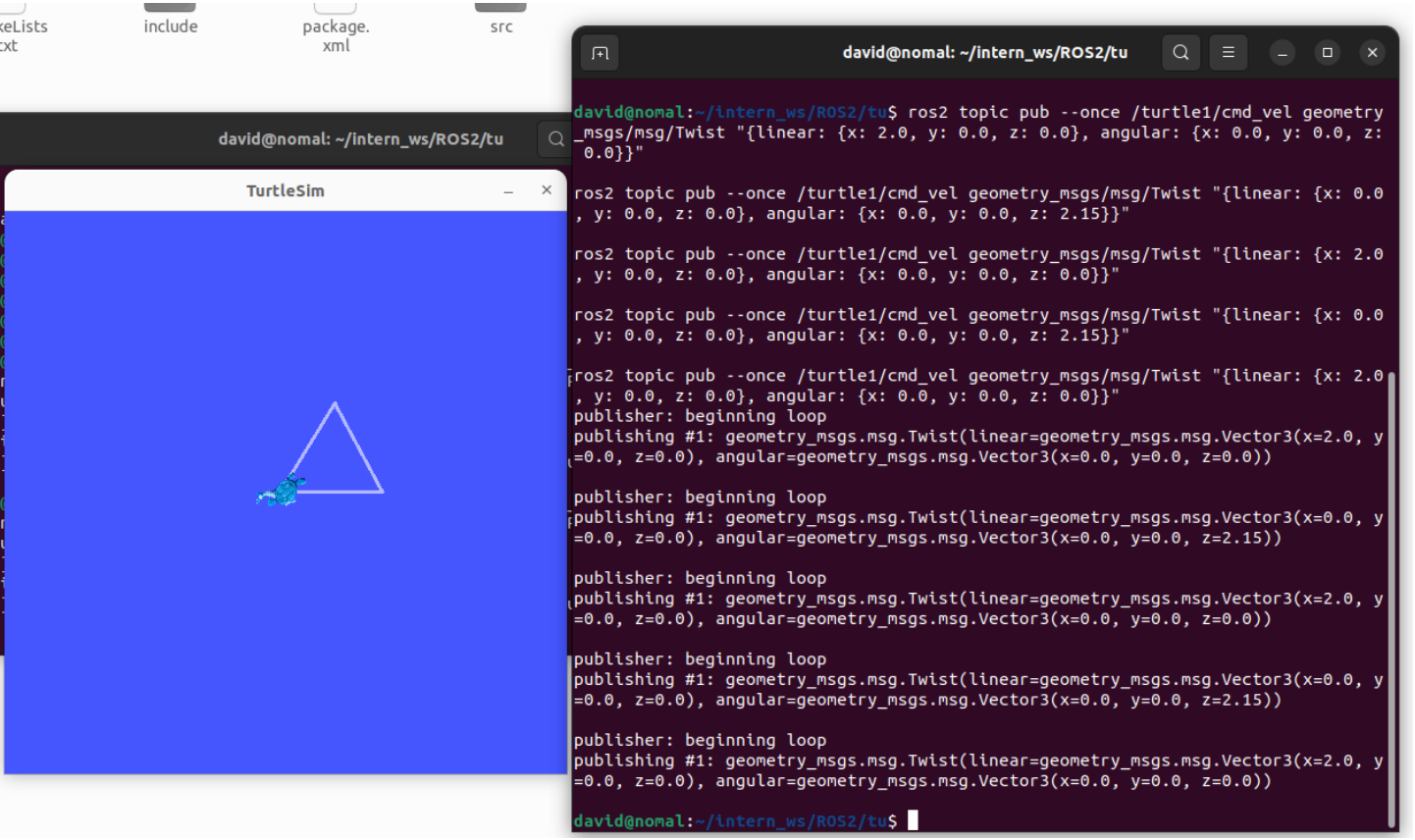
HW1



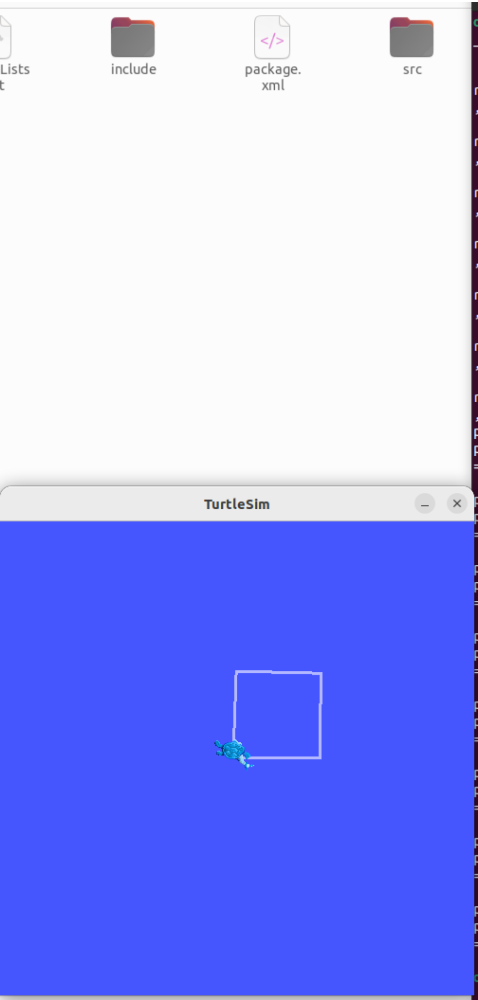
Turtlesim으로 삼각형 그리기

- 2만큼 x 방향으로 전진
- 120도 만큼 왼쪽으로 회전
- 2만큼 x 방향으로 전진
- 120도 만큼 왼쪽으로 회전
- 2만큼 x 방향으로 전진
- 수신자인 turtle1이 행동 수행

HW1

Turtlesim으로 사각형 그리기

- 2만큼 x 방향으로 전진
- 90도 만큼 왼쪽으로 회전

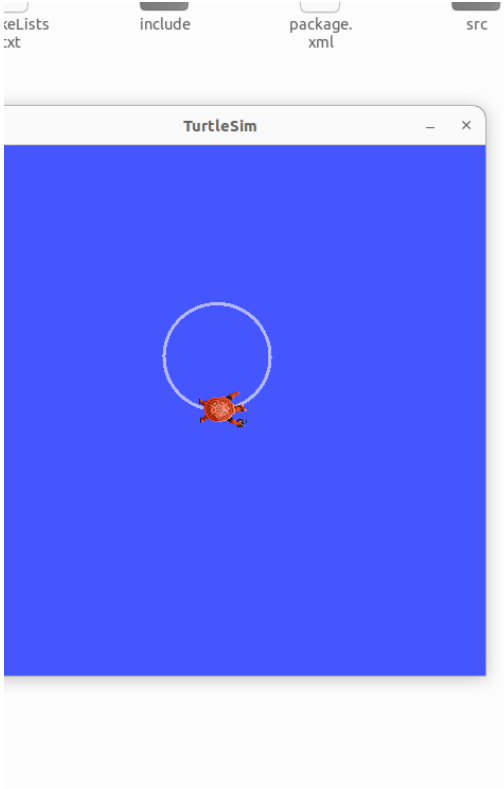


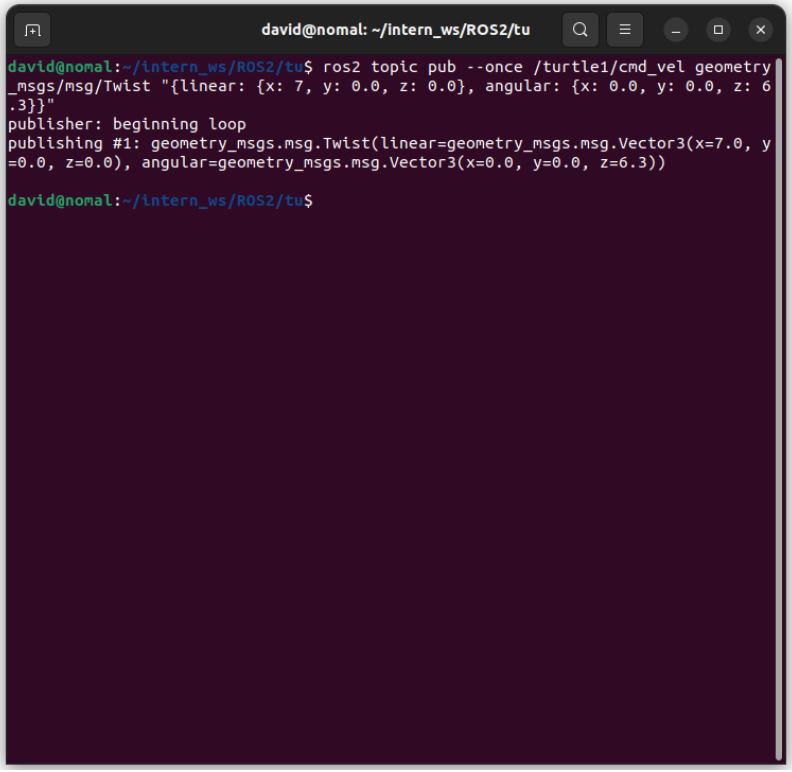

```
david@nomal:~/intern_ws/ROS2/tu$ ros2 topic pub --once /turtle1/cmd_vel geometry
_msgs/msg/Twist "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z:
 0.0}}"

ros2 topic pub --once /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0
, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.624}}"

ros2 topic pub --once /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: 0.0}}"

ros2 topic pub --once /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0
, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.624}}"

ros2 topic pub --once /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: 0.0}}"

ros2 topic pub --once /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0
, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.624}}"

ros2 topic pub --once /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: 0.0}}"

ros2 topic pub --once /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0
, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.624}}"
publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=2.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=0.0))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=0.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=1.624))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=2.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=0.0))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=0.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=1.624))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=2.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=0.0))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=0.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=1.624))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=2.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=0.0))

publisher: beginning loop
publishing #1: geometry_msgs.msg.Twist(linear=geometry_msgs.msg.Vector3(x=0.0, y
=0.0, z=0.0), angular=geometry_msgs.msg.Vector3(x=0.0, y=0.0, z=1.624))

david@nomal:~/intern_ws/ROS2/tu$
```

- 반복

- 수신자인 turtle1이 행동 수행

HW1





Turtlesim으로 원 그리기

- 7 만큼 앞으로 가면서 360도 만큼 회전 하기(중첩)

- 수신자인 turtle1이 행동 수행

(전진과 회전이 중첩되어 원형을 그림)

TOPIC 통신

HW1

- Topic 통신은 노드간 비동기식 단방향 메시지 송수신 방식이다.
- 송신 노드인 Publisher, 수신 노드인 Subscriber가 각각 존재한다.
- 송신 노드와 수신 노드는 각각 특정 대역폭(포트 느낌)을 설정하여 데이터를 주고받는다.
- 특정 대역에서 송신하는 노드의 데이터는 같은 대역을 가진 수신 노드들에게 모두 데이터를 줄 수 있다.
- 따라서 1대1 통신부터 N대N까지도 가능하다.
- 송수신 코드를 모두 넣어 비동기식 양방향 메시지 송수신도 가능하다

Turtlesim

HW1

- ROS에서의 Topic 통신을 사용할 수 있음

```
ros2 topic pub --once /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0
, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.624}}"
```

- Ros2: 명령어,
- Topic: 토픽 통신
- Pub: 송신한다
- --once: 한번만
- Turtle1: turtle1이라는 객체에게
- cmd_vel geometry~~: 해당 명령어에 있는 함수를 사용
- Linear: 직진할 방향과 속도
- Angular: 회전할 축과 회전속도

- 수신 받은 turtle1은 해당 해당 함수에 있는 명령어에 따라 이동 및 회전을 수행한다.

- Topic통신이기에 같은 대역폭을 가진 다른 컴퓨터의 노드도 해당 명령어를 받을 수 있다.

- 다른 친구의 신호가 들어온 상황

TOPIC 통신 활용

HW1

- 센서 값 입력 받기
- 단체 행동 로봇
- 단체 채팅