ROS2 – 9 เขียน Node ภาษา Python แบบ OOP



เขียน Node ภาษา Python แบบ OOP

การเขียนโปรแกรมเชิงวัตถุ OOP (Object Oriented Programming) เป็นวิธีการเขียนโปรแกรมที่นำแนวคิดในโลกของความเป็นจริงมาใช้กับในโลกของการเขียนโปรแกรมในการที่ โปรแกรมเมอร์ สร้างซอฟต์แวร์ขึ้นมาเพื่อแก้ไขปัญหาต่างๆ OOP เป็นวิธีการเขียนโปรแกรมรูปแบบหนึ่ง โดยมองสิ่งต่างๆในระบบเป็นวัตถุ (Object) ชิ้นหนึ่งที่มีหน้าที่และความหมายในตัว โดยวัตถุๆนั้น ก็มี คุณสมบัติ (Attributes) และ พฤติกรรม (Method,Behavior) หรือการกระทำของมัน เป็นการมองบนพื้นฐานความเป็นจริงมากขึ้น

คำศัพท์ที่จำเป็นต้องทราบสำหรับการเขียนโปรแกรมเชิงวัตถุในภาษา Python

* คลาส คือประเภทข้อมูลที่สร้างโดยผู้ใช้ โดยจะนำไปใช้สร้างออบเจ็ค กล่าวอีกนัยหนึ่ง คลาสคือประเภทข้อมูลของออบเจ็ค
* ออบเจ็ค คือสิ่งที่สร้างมาจากคลาสหรือ class instances
* แอตทริบิวต์ (instance attributes) คือข้อมูลที่เป็นสมาชิกของแต่ละออบเจ็ค โดยมักจะกำหนดไว้ในเมธอด **init**() ของคลาส
* เมธอด คือฟังก์ชันการทำงานที่กำหนดไว้ในคลาส
* คลาสแอตทริบิวต์ (class attributes) คือตัวแปรที่ประกาศไว้ในคลาส ซึ่งจะแชร์กับออบเจ็คทั้งหมดที่สร้างจากคลาสนั้นๆ

**1 : เขียน Node ภาษา Python เบื้องต้น**  
เขียน Node ภาษา Python เบื้องต้น ตามขั้นตอนลิงค์ด้านล่าง

* [เขียน Node ภาษา Python เบื้องต้น](http://www.lungmaker.com/ros2-8-%e0%b9%80%e0%b8%82%e0%b8%b5%e0%b8%a2%e0%b8%99-node-%e0%b8%a0%e0%b8%b2%e0%b8%a9%e0%b8%b2-python-%e0%b9%80%e0%b8%9a%e0%b8%b7%e0%b9%89%e0%b8%ad%e0%b8%87%e0%b8%95%e0%b9%89%e0%b8%99/)

**2 : เขียน Node ภาษา Python แบบ OOP**

ใช้ Visual Studio Code เปิดไฟล์ my\_first\_node.py เขียนโค้ด + Save ตามโค้ดด้านล่าง

#!/usr/bin/env python3

import rclpy

from rclpy.node import Node

class MyNode(Node):

def \_\_init\_\_(self):

super().\_\_init\_\_("py\_test")

self.get\_logger().info("Hello ROS2")

def main(args=None):

rclpy.init(args=args)

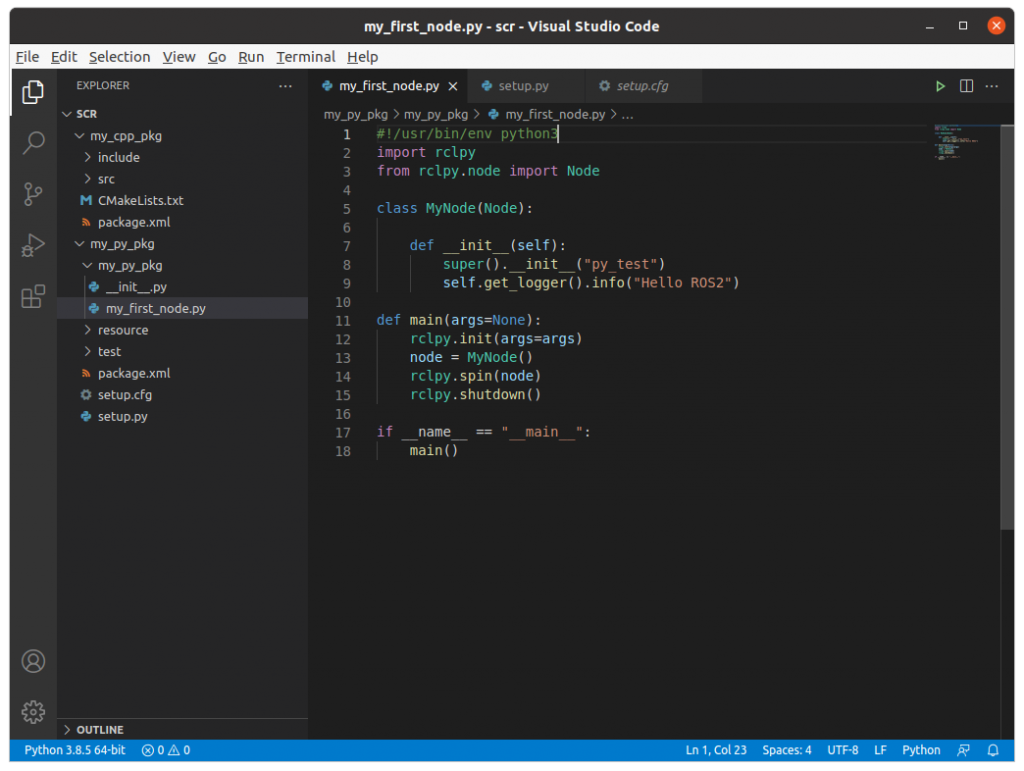
node = MyNode()

rclpy.spin(node)

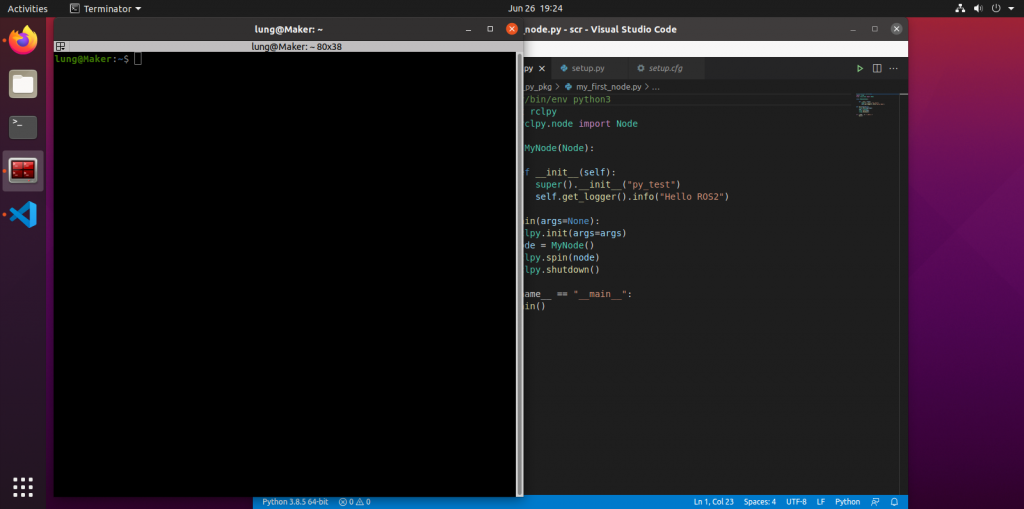
rclpy.shutdown()

if \_\_name\_\_ == "\_\_main\_\_":

main()

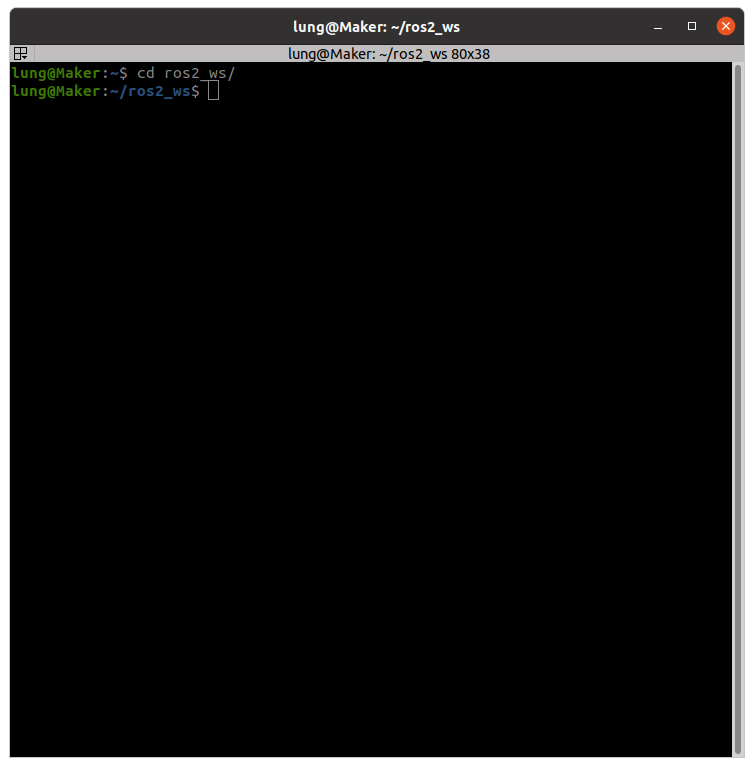


เริ่มการทำงานโดย เปิด Terminator



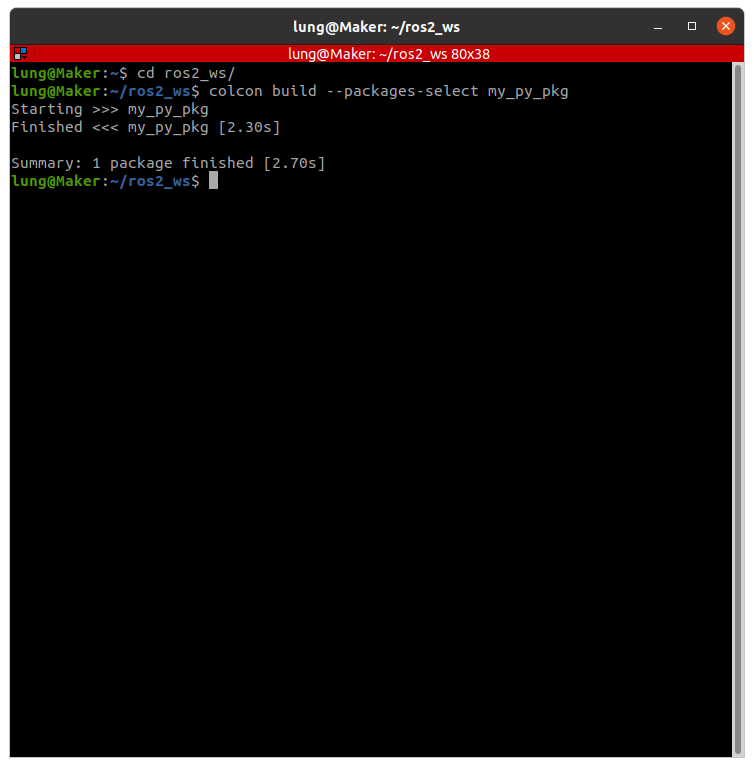
เข้าไปใน โฟลเดอร์ ros2\_ws

cd ros2\_ws/



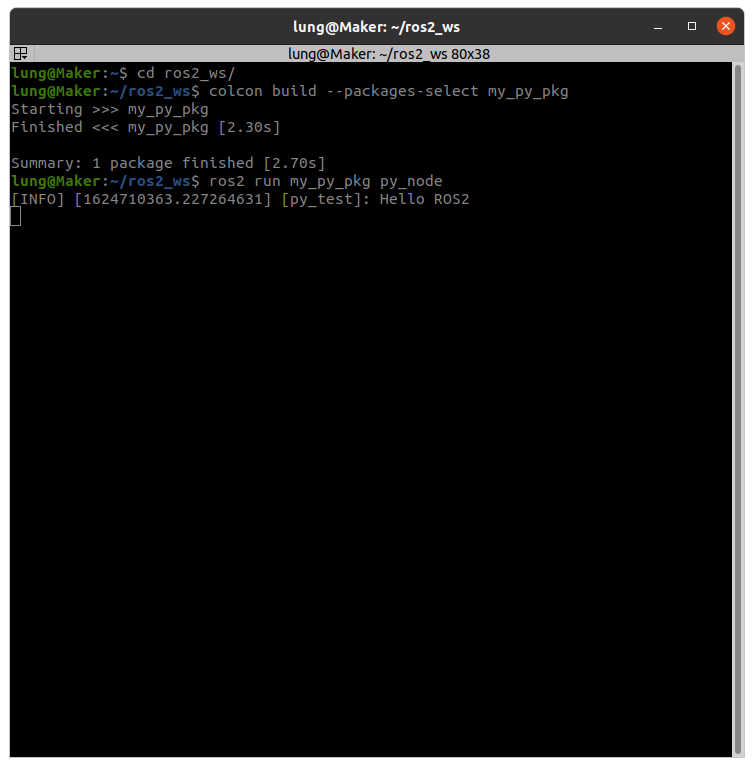
Build แพ็คเกจ my\_py\_pkg

colcon build --packages-select my\_py\_pkg

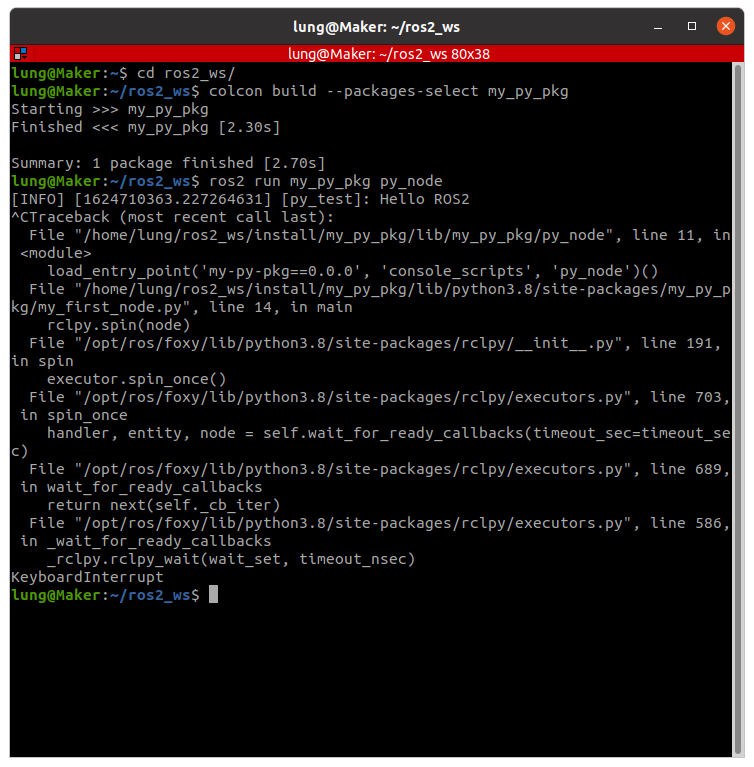


เรียกใช้งานด้วยคำสั่งของ ros2 จะได้ผลลัพธ์ การทำงาน เหมือนกับ การเขียน Node ภาษา Python เบื้องต้น

ros2 run my\_py\_pkg py\_node



กด Ctrl + c เพื่อหยุดการทำงาน



**3 : เพิ่ม ฟังก์ชั่น timer\_callback**  
ใช้ Visual Studio Code เปิดไฟล์ my\_first\_node.py เขียนโค้ด + Save ตามโค้ดด้านล่าง

#!/usr/bin/env python3

import rclpy

from rclpy.node import Node

class MyNode(Node):

def \_\_init\_\_(self):

super().\_\_init\_\_("py\_test")

self.counter\_ = 0

self.get\_logger().info("Hello ROS2")

self.create\_timer(0.5, self.timer\_callback)

def timer\_callback(self):

self.counter\_ += 1

self.get\_logger().info("Hello" + str(self.counter\_))

def main(args=None):

rclpy.init(args=args)

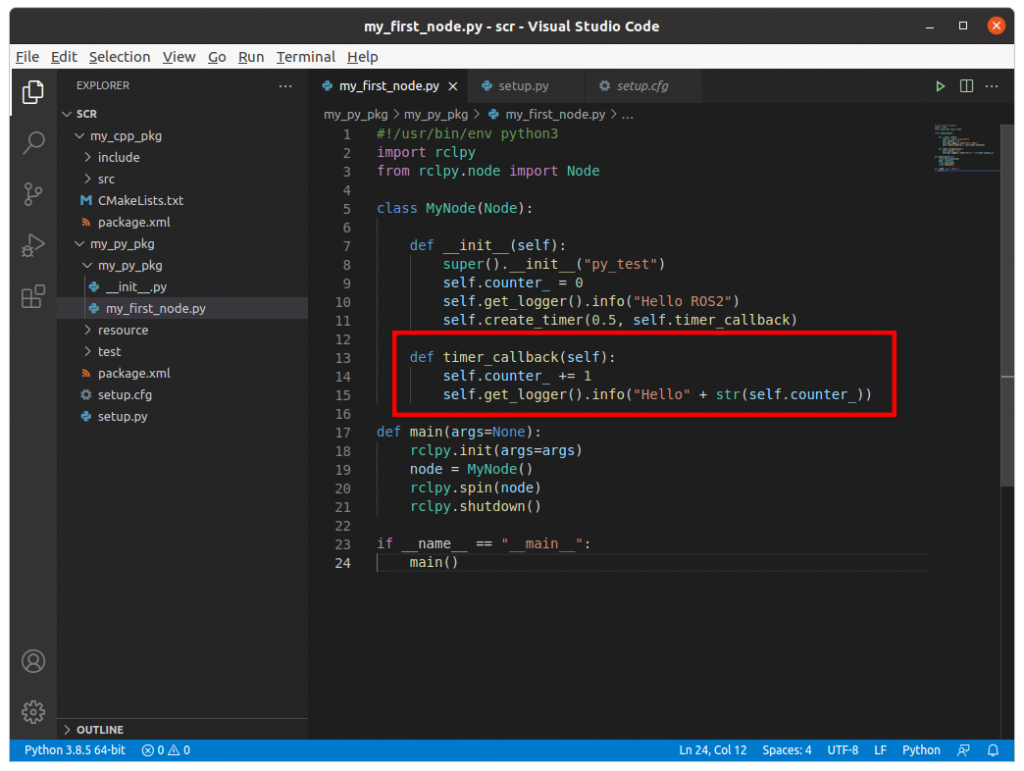
node = MyNode()

rclpy.spin(node)

rclpy.shutdown()

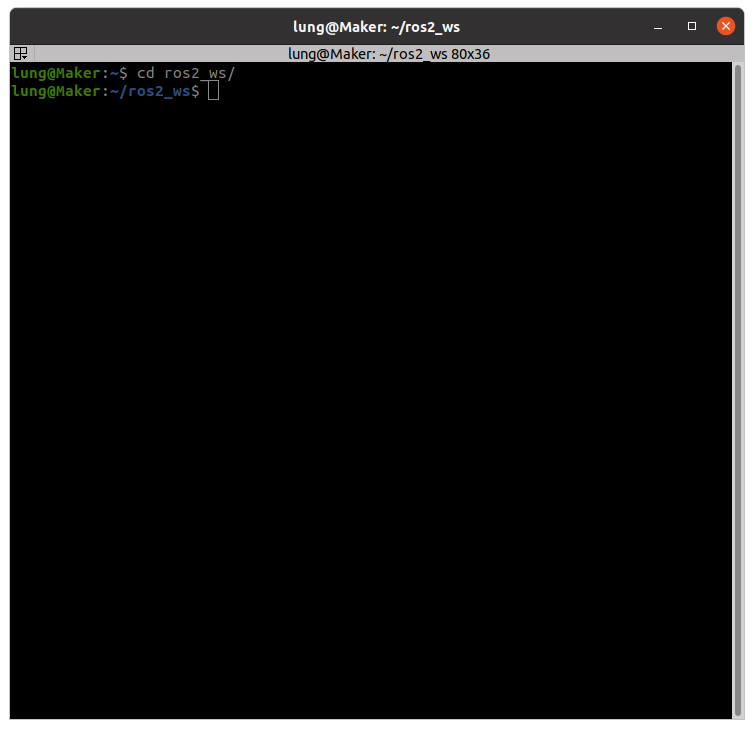
if \_\_name\_\_ == "\_\_main\_\_":

main()



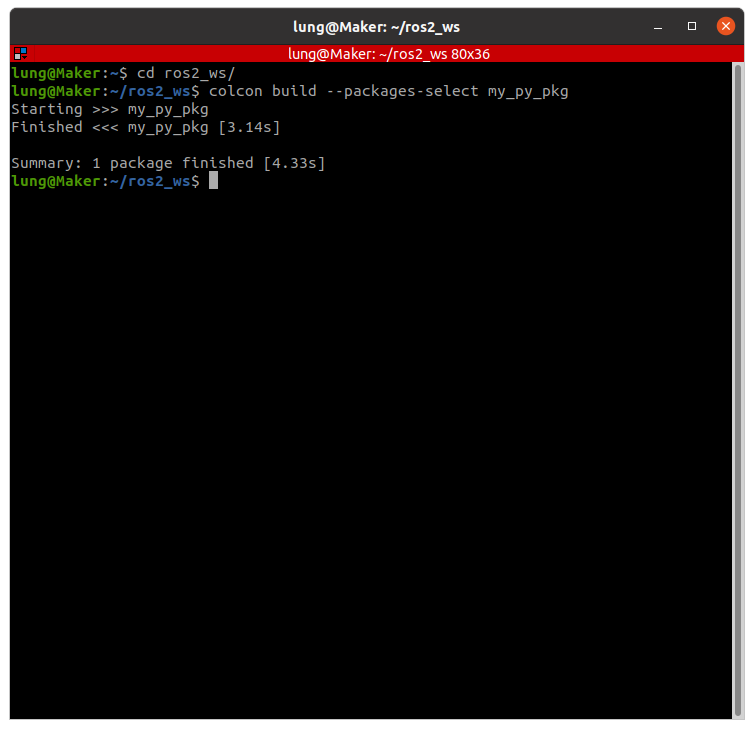
เปิด Terminator เข้าไปใน โฟลเดอร์ ros2\_ws

cd ros2\_ws/

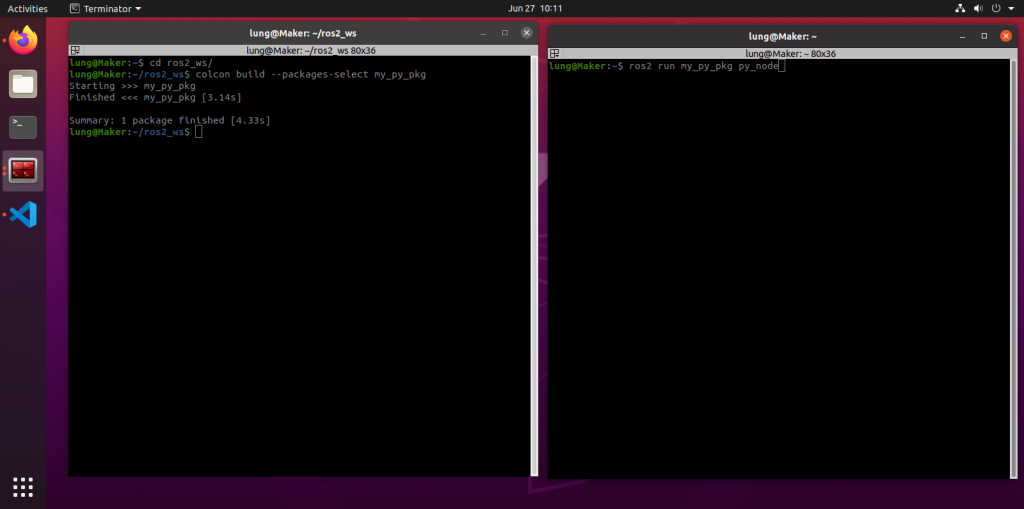


Build แพ็คเกจ my\_py\_pkg

colcon build --packages-select my\_py\_pkg

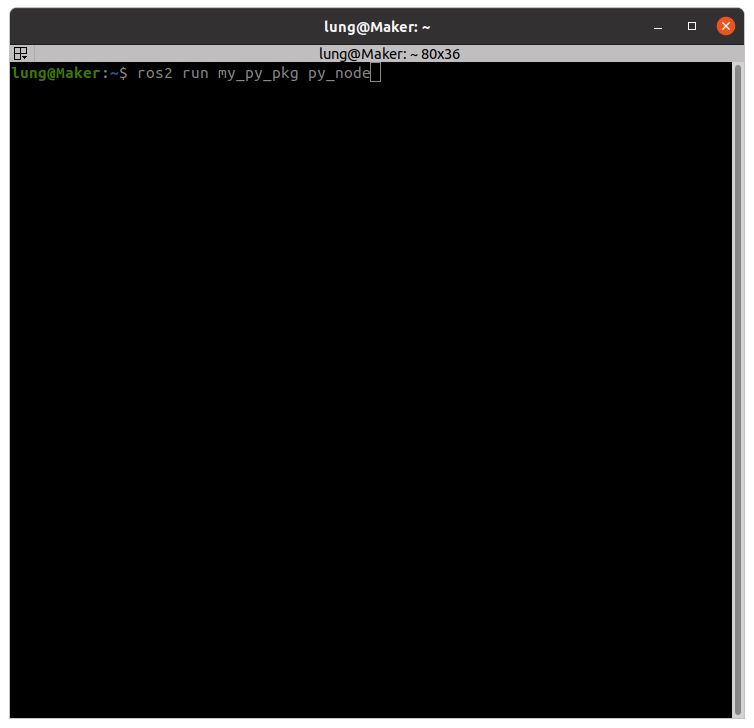


เปิด Terminator หน้าต่างที่ 2



เรียกใช้งานด้วยคำสั่งของ ros2

ros2 run my\_py\_pkg py\_node



จะได้ผลลัพธ์ การทำงานตามรูปด้านล่าง

