ROS2 – 14 เขียน Publisher / Subscriber ด้วยภาษา C++



เขียน Publisher และ Subscriber ด้วยภาษา C++

Topics : ข้อความจะถูกส่งผ่านระบบการขนส่งที่มีการ Publisher / Subscriber โหนดที่ส่งข้อความจะเรียกว่า Publisher ชื่อที่ระบุใน Topic จะใช้เพื่อระบุเนื้อหาของข้อความ โหนดที่สนใจในข้อมูลบางประเภทจะทำการ Subscriber หัวข้อที่ต้องการ อาจมีการ Publisher / Subscribe หลายรายพร้อมกันสำหรับหัวข้อเดียวและโหนดเดียวอาจ Publisher / Subscriber ได้หลาย Topics โดยทั่วไป Publisher / Subscriber จะไม่ได้ตระหนักถึงการมีอยู่ของกันและกัน กล่าวอีกนัยคือการแยกกันผลิตข้อมูลตามหน้าที่ของตนเอง

Topic ก็เหมือนกับหัวเรื่องที่เราไว้คุยกัน พูดง่ายๆคือ Publisher Node จะ Publish Message ของตัวเองขึ้นไปที่ Topic แล้วจะมี Subscriber Node มารอรับ Message ที่ส่งมาทาง Topic ที่มีชื่อเดียวกัน

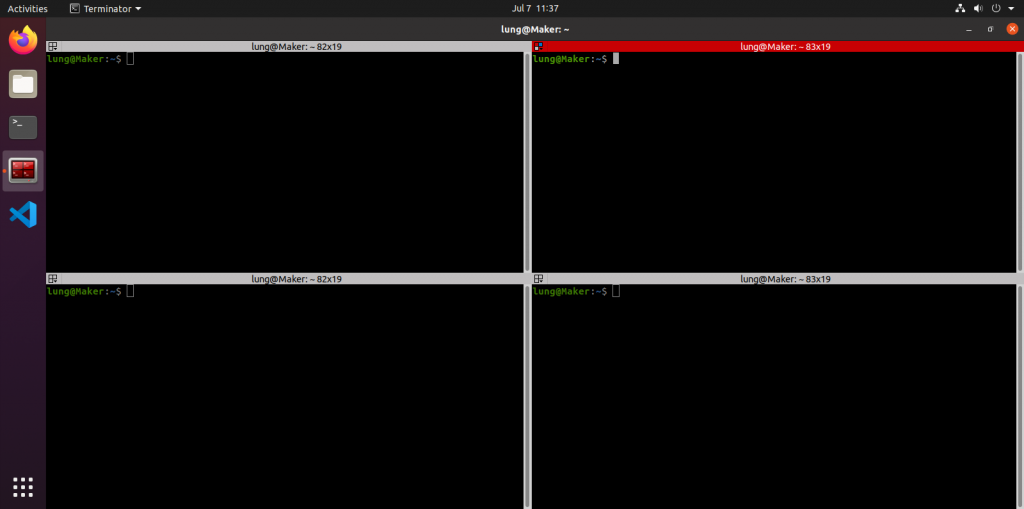
เวลาพูดว่า Publisher หมายถึงการที่ Node จะส่ง Message ออกไปที่ Topic โดยไม่สนใจว่าจะมี Node รอรับอยู่รึเปล่า Node 1 ตัวสามารถ Publisher Message ออกไปได้หลายๆ Topics

Message : ข้อความ ค่ามาตรฐาน (จำนวนเต็ม, จุดลอย, บูลีน, ฯลฯ ) ข้อมูลอาร์เรย์ทั่วๆไป รวมไปถึงข้อมูลอาร์เรย์ที่มีความซับซ้อน

**1 : เขียน Node ภาษา C++ แบบ OOP**  
เขียน Node ภาษา C++ แบบ OOP ตามขั้นตอนลิงค์ด้านล่าง

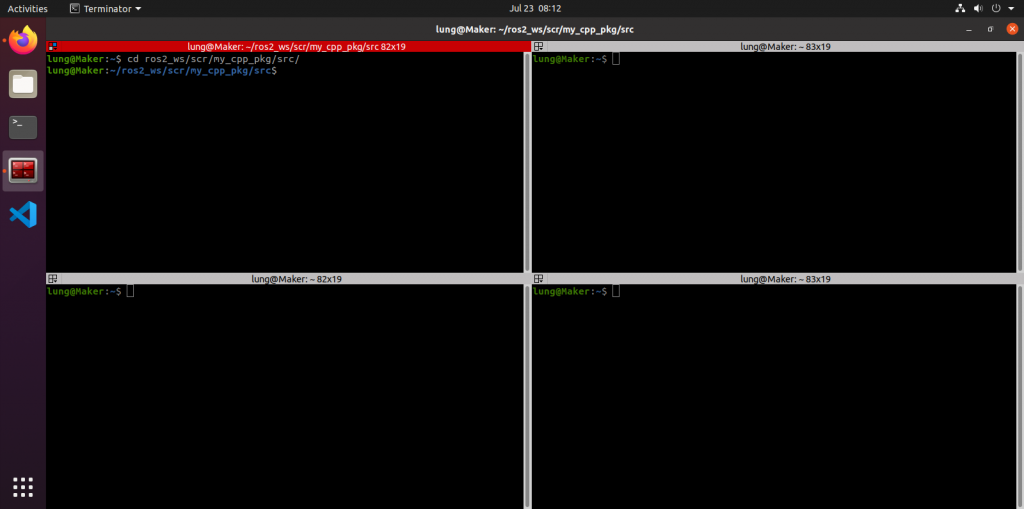
* [เขียน Node ภาษา C++ แบบ OOP](http://www.lungmaker.com/ros2-11-%e0%b9%80%e0%b8%82%e0%b8%b5%e0%b8%a2%e0%b8%99-node-%e0%b8%a0%e0%b8%b2%e0%b8%a9%e0%b8%b2-c-%e0%b9%81%e0%b8%9a%e0%b8%9a-oop/)

**2 : เปิด Terminator 4 หน้าต่าง**

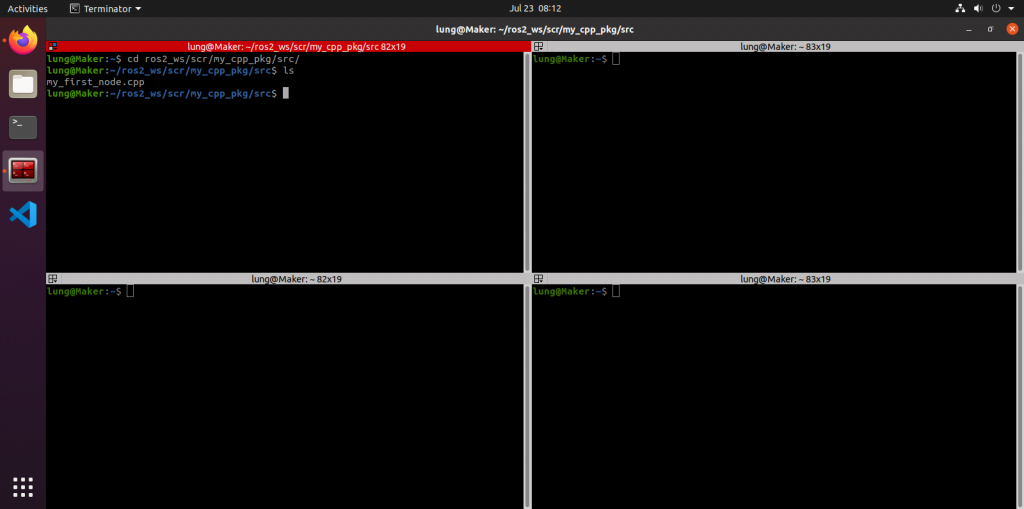


**3 : เขียน Publisher ด้วยภาษา C++**  
เขียนโหนดส่งข้อความ Publisher ด้วยภาษา C++  
โดยไปที่ หน้าต่างที่ 1 ใช้คำสั่งเข้าไปใน โฟลเดอร์ที่ต้องการสร้างไฟล์

cd ros2\_ws/scr/my\_cpp\_pkg/src/

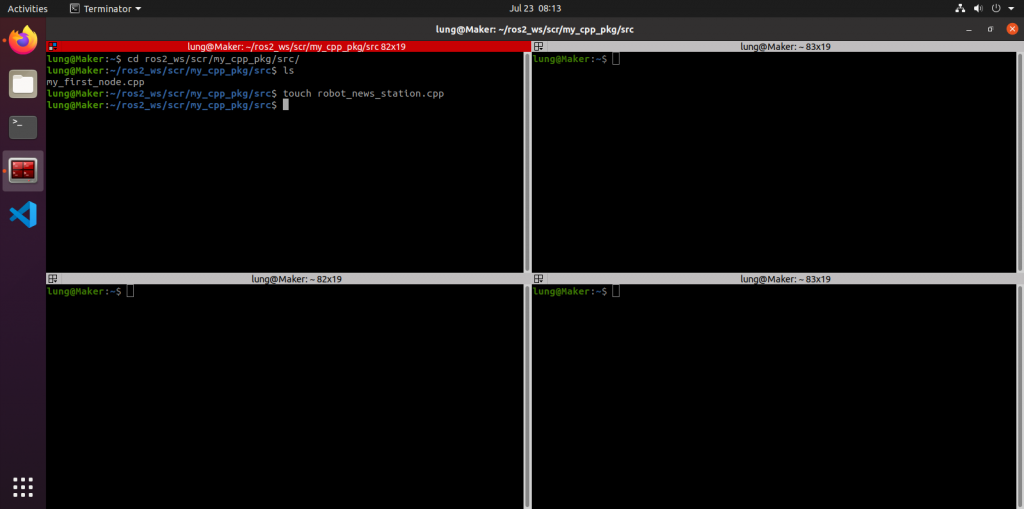


ls



สร้างไฟล์ robot\_news\_station.cpp

touch robot\_news\_station.cpp



ใช้ Visual Studio Code เปิดไฟล์ robot\_news\_station.cpp เขียนโค้ด + Save ตามโค้ดด้านล่าง

#include "rclcpp/rclcpp.hpp"

#include "example\_interfaces/msg/string.hpp"

class RobotNewsStationNode : public rclcpp::Node

{

public:

RobotNewsStationNode() : Node("robot\_news\_station"), robot\_name\_("R2D2")

{

publisher\_ = this->create\_publisher<example\_interfaces::msg::String>("robot\_news", 10);

timer\_ = this->create\_wall\_timer(std::chrono::milliseconds(500),

std::bind(&RobotNewsStationNode::publishNews, this));

RCLCPP\_INFO(this->get\_logger(), "Robot News Station has been started.");

}

private:

void publishNews()

{

auto msg = example\_interfaces::msg::String();

msg.data = std::string("Hi, this is ") + robot\_name\_ + std::string(" from the Robot News Station");

publisher\_->publish(msg);

}

std::string robot\_name\_;

rclcpp::Publisher<example\_interfaces::msg::String>::SharedPtr publisher\_;

rclcpp::TimerBase::SharedPtr timer\_;

};

int main(int argc, char \*\*argv)

{

rclcpp::init(argc, argv);

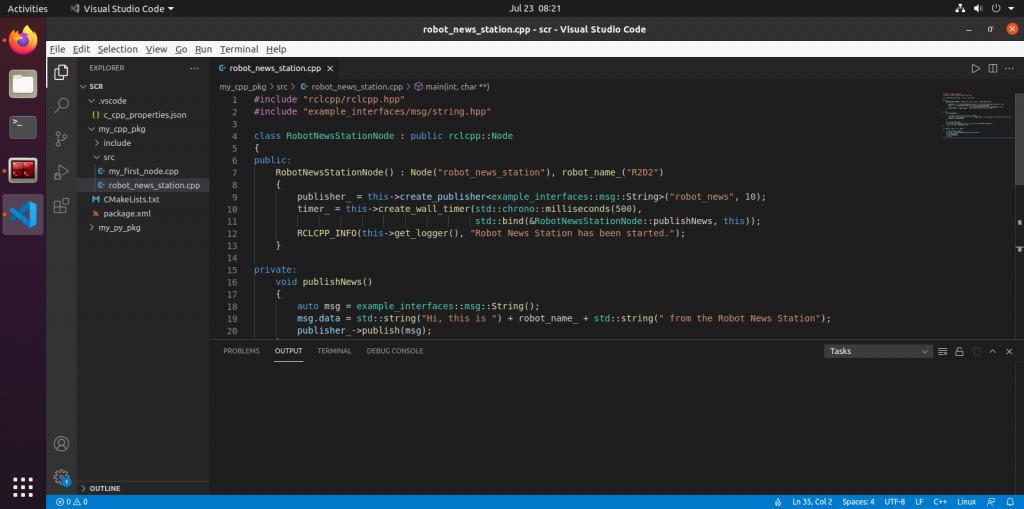
auto node = std::make\_shared<RobotNewsStationNode>();

rclcpp::spin(node);

rclcpp::shutdown();

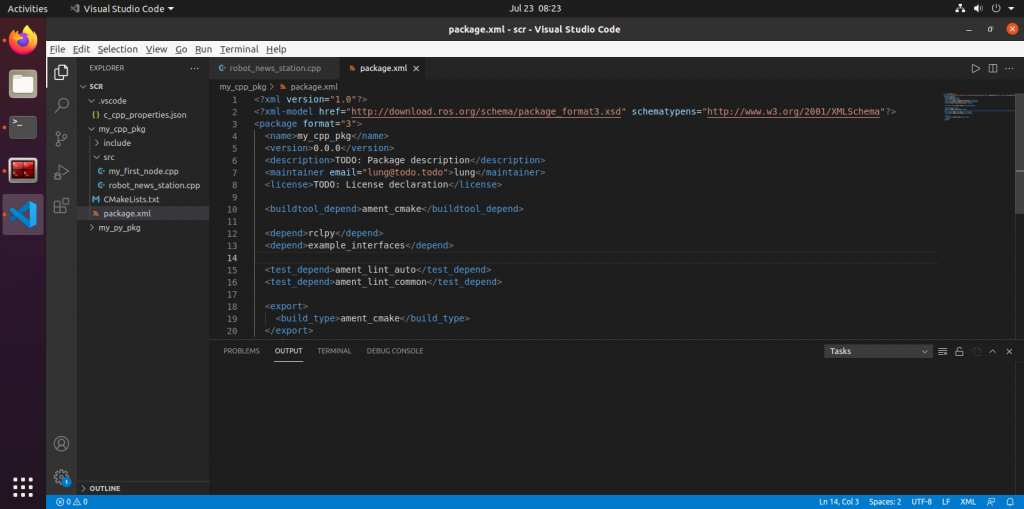
return 0;

}



เปิดไฟล์ package.xml เพิ่มโค้ดบรรทัดที่ 13 ด้วยโค้ดด้านล่าง

<depend>example\_interfaces</depend>



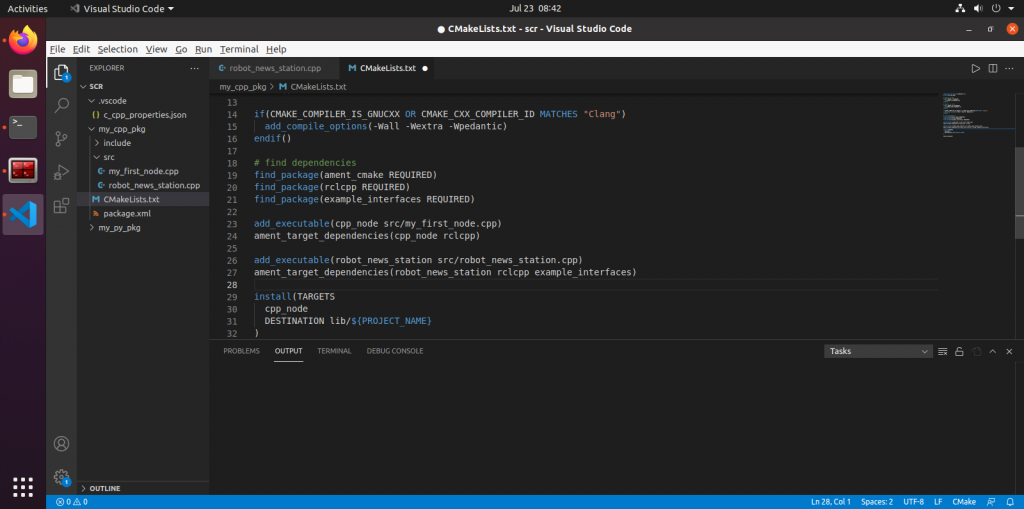
เปิดไฟล์ CMakeLists.txt เพิ่มโค้ดบรรทัดที่ 21 ด้วยโค้ดด้านล่าง

find\_package(example\_interfaces REQUIRED)

และ เพิ่มโค้ดบรรทัดที่ 26 และ 27 ด้วยโค้ดด้านล่าง

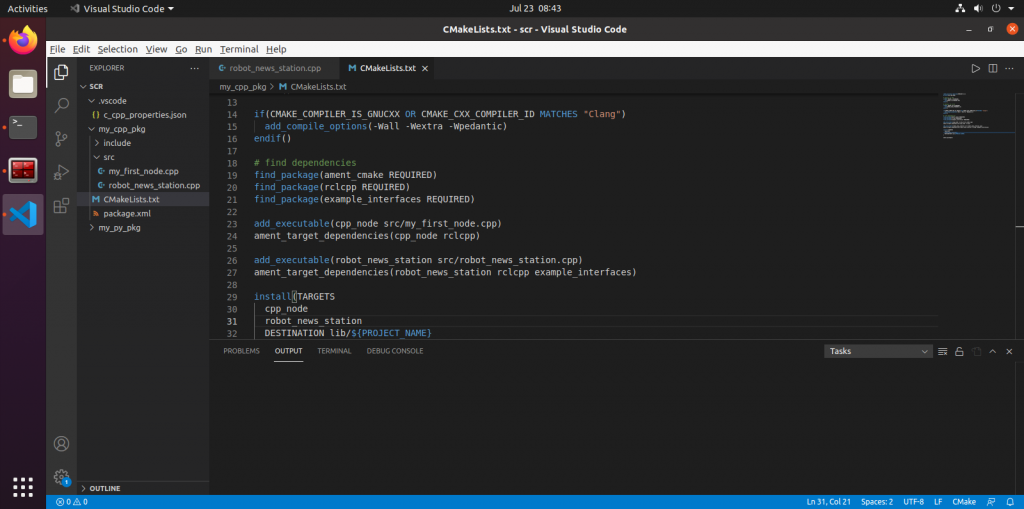
add\_executable(robot\_news\_station src/robot\_news\_station.cpp)

ament\_target\_dependencies(robot\_news\_station rclcpp example\_interfaces)



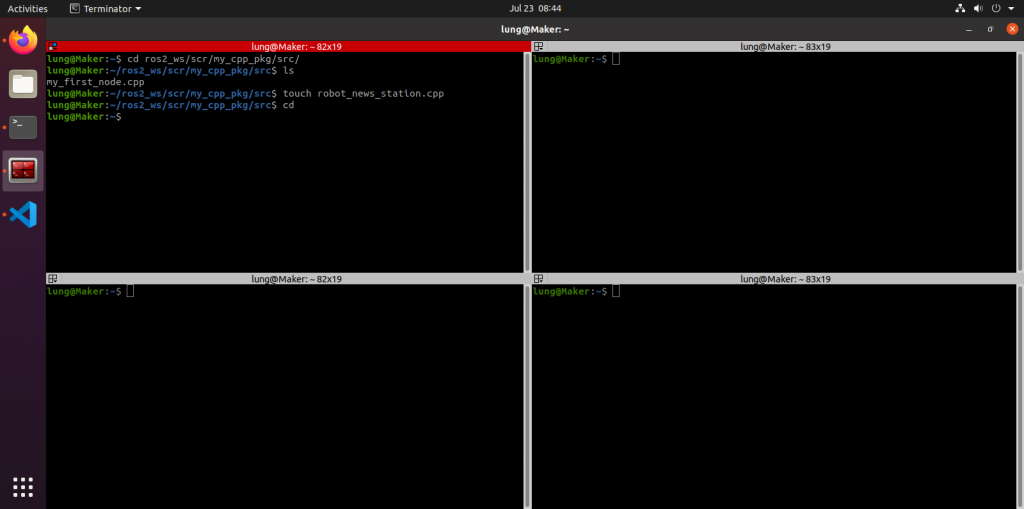
เพิ่มโค้ดบรรทัดที่ 31 ด้วยโค้ดด้านล่าง

robot\_news\_station

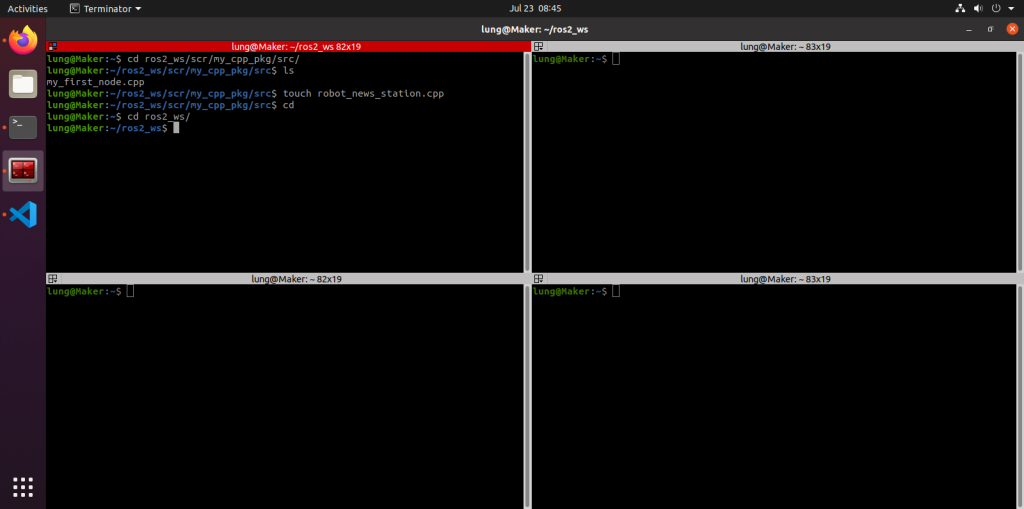


กลับสู่ Home เพื่อจะ Build แพ็กเกจ my\_cpp\_pkg

cd

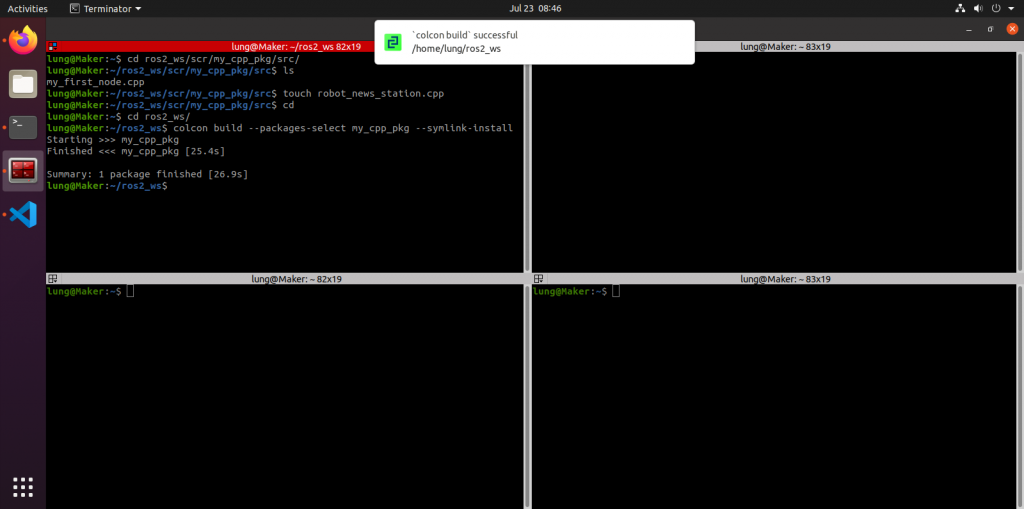


cd ros2\_ws/



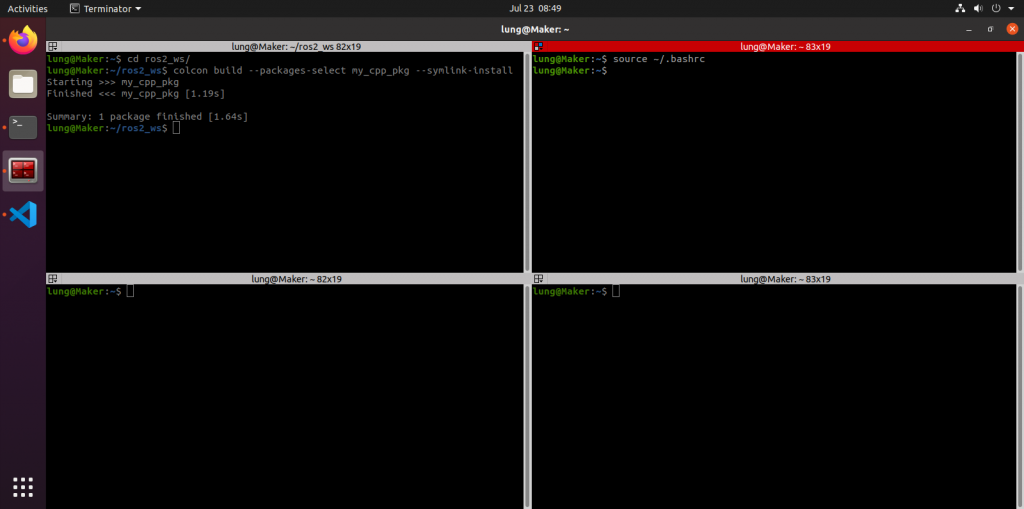
คำสั่ง Build เฉพาะ แพ็กเกจ my\_cpp\_pkg

colcon build --packages-select my\_cpp\_pkg --symlink-install



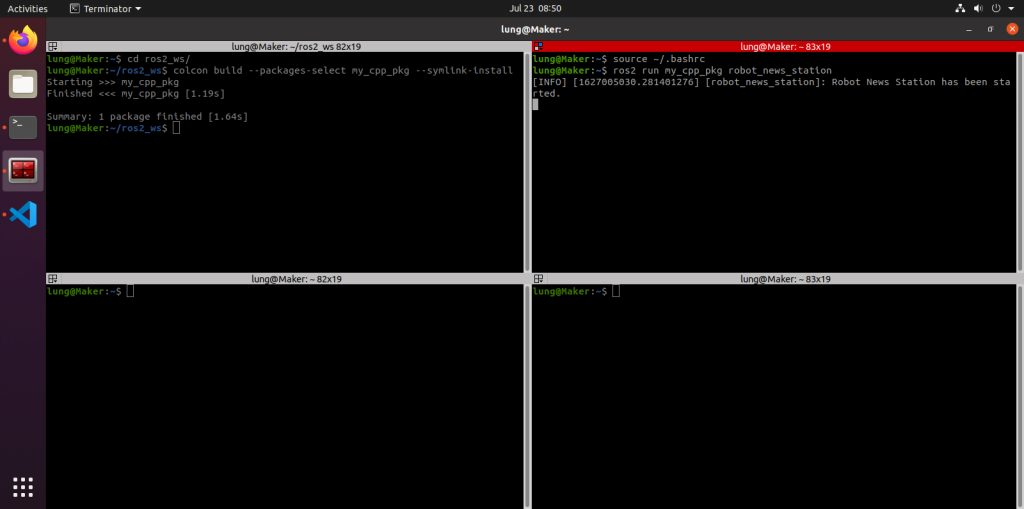
ไปที่ หน้าต่างที่ 2 โดยเราจะต้องทําให้ Terminator รู้จักคําสั่งของ ROS2 ด้วย คําสั่ง

source ~/.bashrc



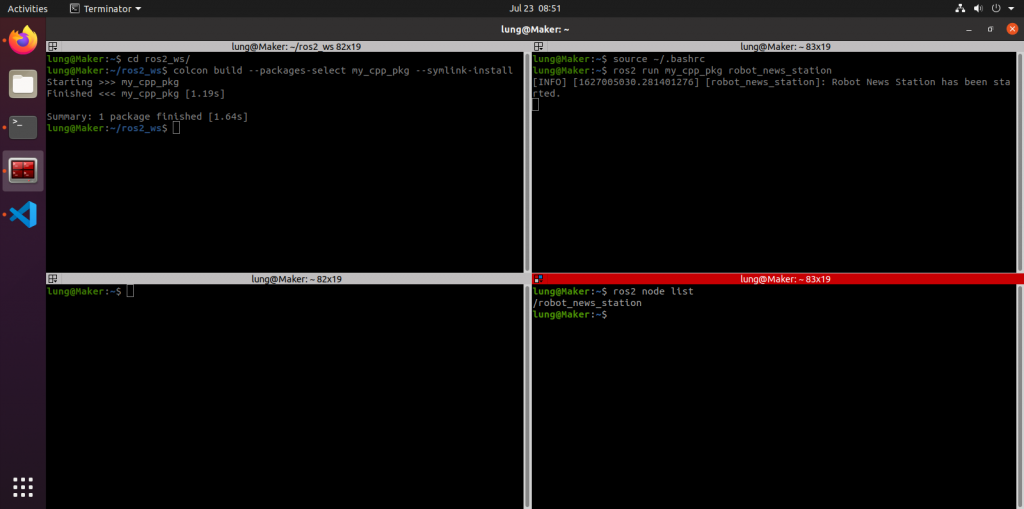
เรียกใช้งาน robot\_news\_station ด้วย ROS2

ros2 run my\_cpp\_pkg robot\_news\_station



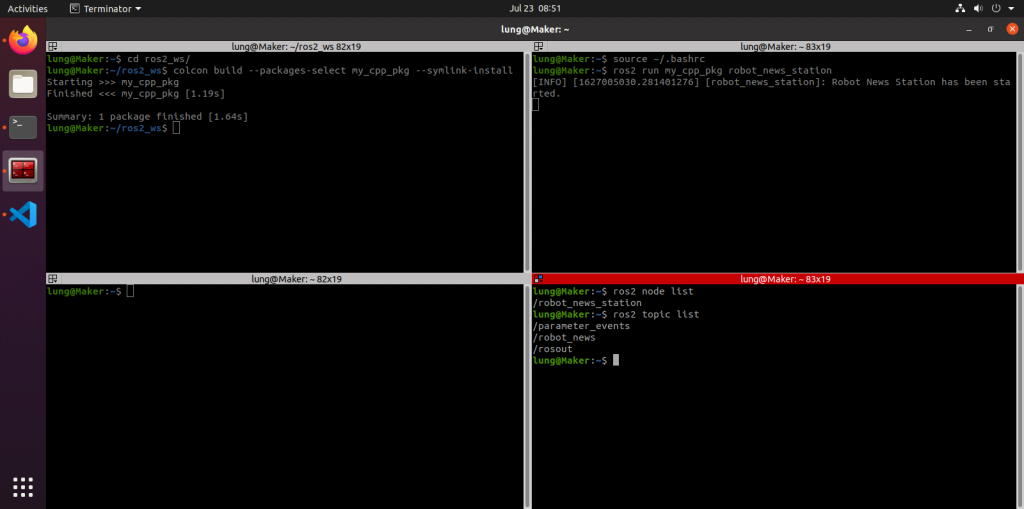
ไปที่ หน้าต่างที่ 4 ให้แสดง Node ที่กำลังทำงานอยู่

ros2 node list



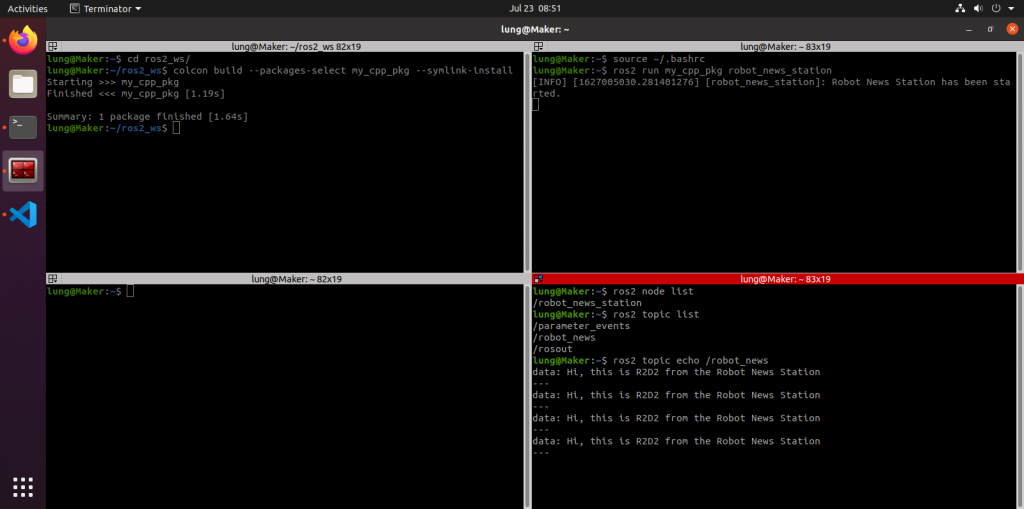
แสดง Topic ที่กำลังทำงานอยู่

ros2 topic list



แสดงข้อมูลที่ส่งออกมาจาก topic robot\_news

ros2 topic echo /robot\_news

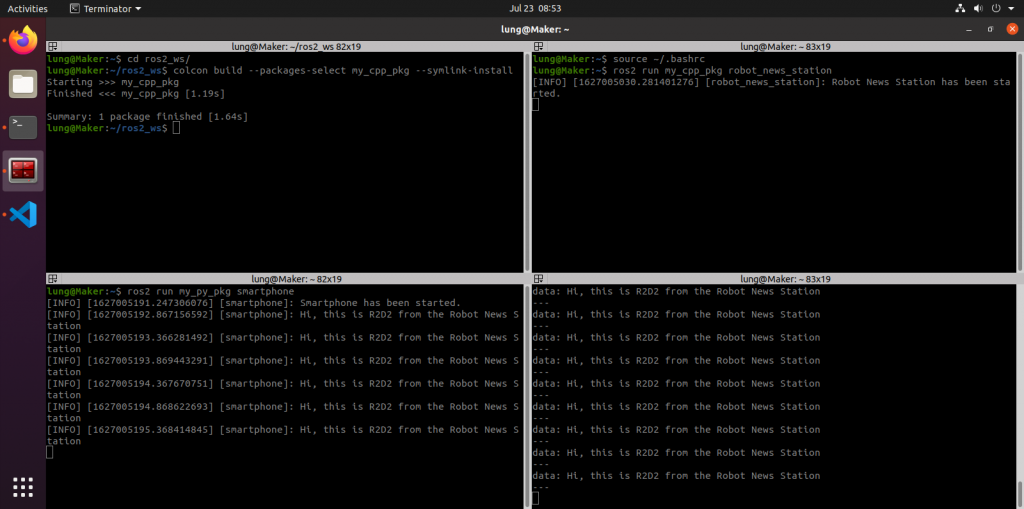


ไปที่ หน้าต่างที่ 3 เรียกใช้งาน smartphone ที่เขียนด้วยภาษา Python ก่อนหน้านี้

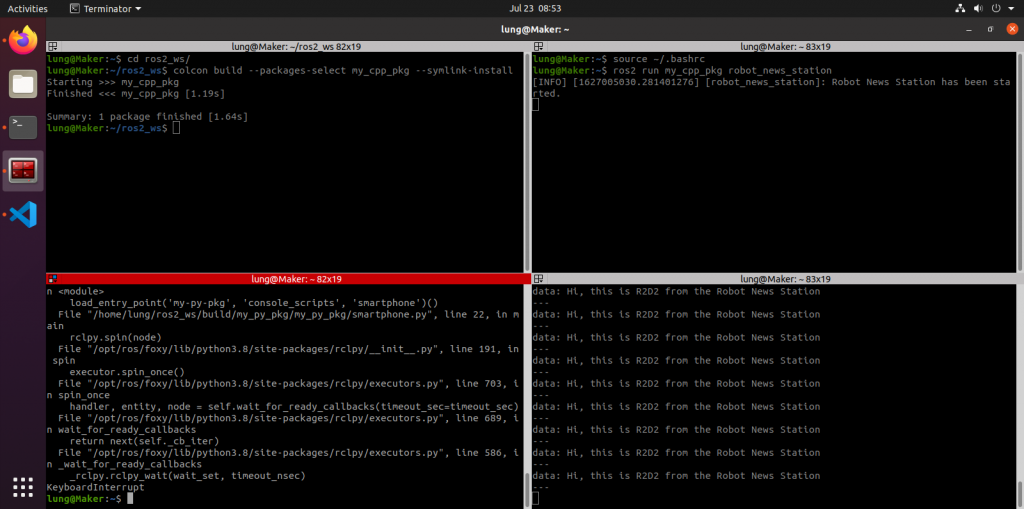
* [เขียน Publisher / Subscriber ด้วยภาษา Python](http://www.lungmaker.com/ros2-13-%e0%b9%80%e0%b8%82%e0%b8%b5%e0%b8%a2%e0%b8%99-publisher-%e0%b8%94%e0%b9%89%e0%b8%a7%e0%b8%a2%e0%b8%a0%e0%b8%b2%e0%b8%a9%e0%b8%b2-python/)

จะแสดงการรับ Message จากที่ส่งออกมาจาก topic robot\_news ของภาษา C++ ซึ่งแสดงให้เห็นว่า ROS2 สามารถทำงานข้ามภาษาได้

ros2 run my\_py\_pkg smartphone



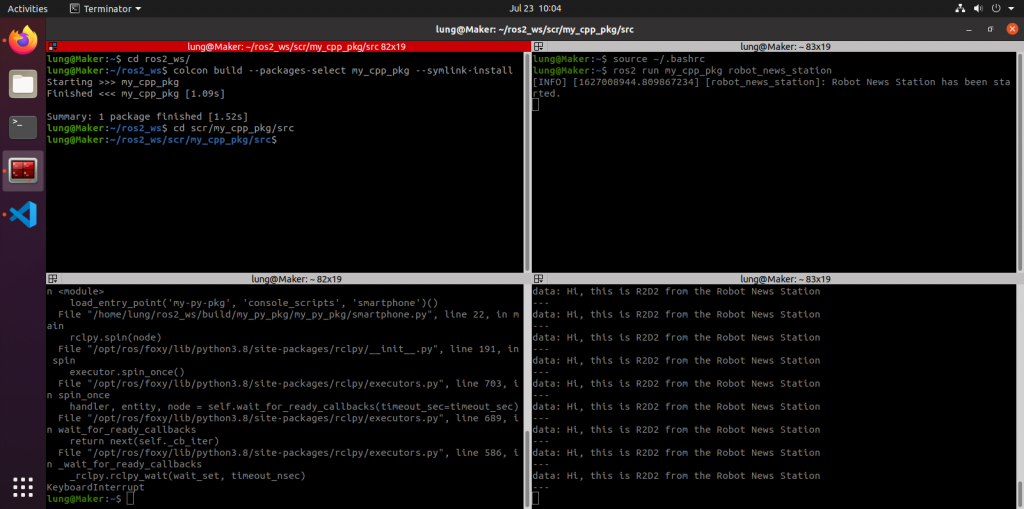
กด Ctrl + c เพื่อหยุดการทำงาน



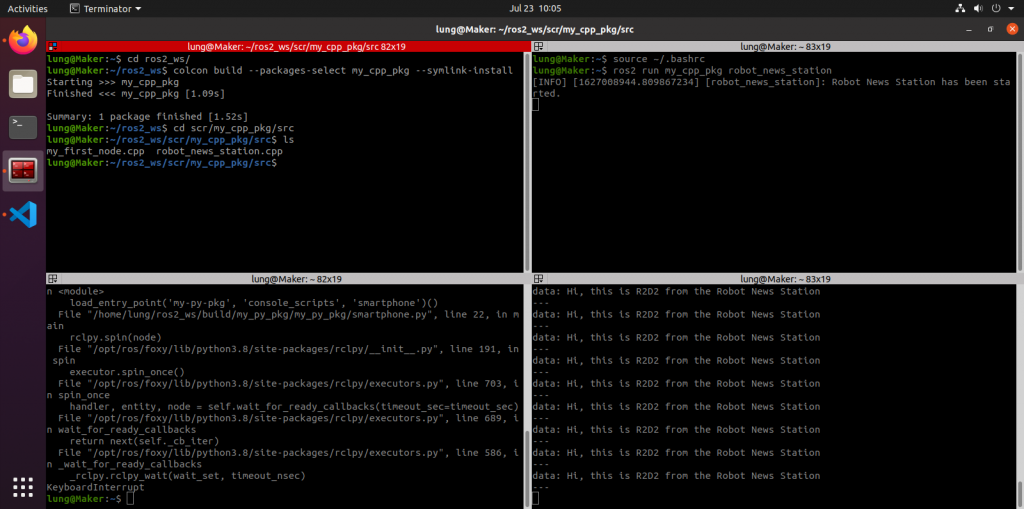
**4 : เขียน Subscriber ด้วยภาษา C++**  
Subscriber หมายถึงการที่ Node จะรับ Message จาก Topic ที่ตัวเองรอรับอยู่ Node 1 ตัวสามารถ Subscriber Message ที่มาจากหลายๆ Topic ได้

เขียน โหนด รับข้อความ Subscriber ด้วยภาษา C++ โดยกลับไปที่ หน้าต่างที่ 1  ใช้คำสั่งเข้าไปใน โฟลเดอร์ที่ต้องการสร้างไฟล์

cd scr/my\_cpp\_pkg/src

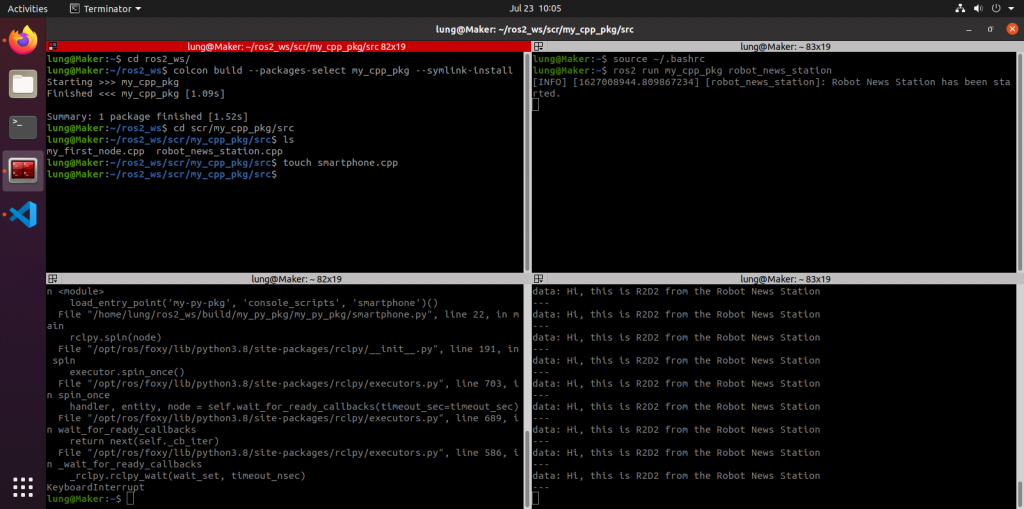


ls



สร้างไฟล์ smartphone.cpp

touch smartphone.cpp



ใช้ Visual Studio Code เปิดไฟล์ smartphone.cpp เขียนโค้ด + Save ตามโค้ดด้านล่าง

#include "rclcpp/rclcpp.hpp"

#include "example\_interfaces/msg/string.hpp"

class SmartphoneNode : public rclcpp::Node

{

public:

SmartphoneNode() : Node("smartphone")

{

subscriber\_ = this->create\_subscription<example\_interfaces::msg::String>(

"robot\_news", 10,

std::bind(&SmartphoneNode::callbackRobotNews, this, std::placeholders::\_1));

RCLCPP\_INFO(this->get\_logger(), "Smartphone has been started.");

}

private:

void callbackRobotNews(const example\_interfaces::msg::String::SharedPtr msg)

{

RCLCPP\_INFO(this->get\_logger(), "%s", msg->data.c\_str());

}

rclcpp::Subscription<example\_interfaces::msg::String>::SharedPtr subscriber\_;

};

int main(int argc, char \*\*argv)

{

rclcpp::init(argc, argv);

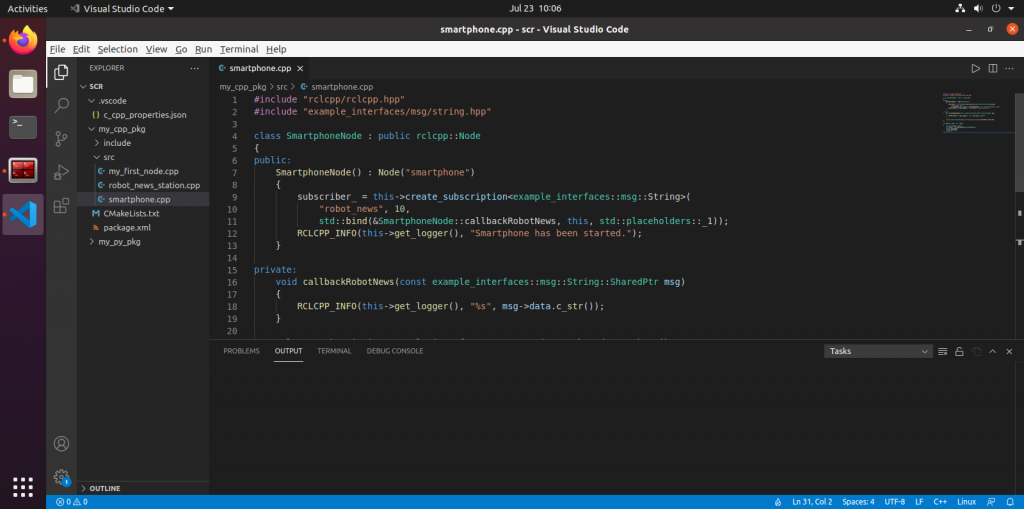
auto node = std::make\_shared<SmartphoneNode>();

rclcpp::spin(node);

rclcpp::shutdown();

return 0;

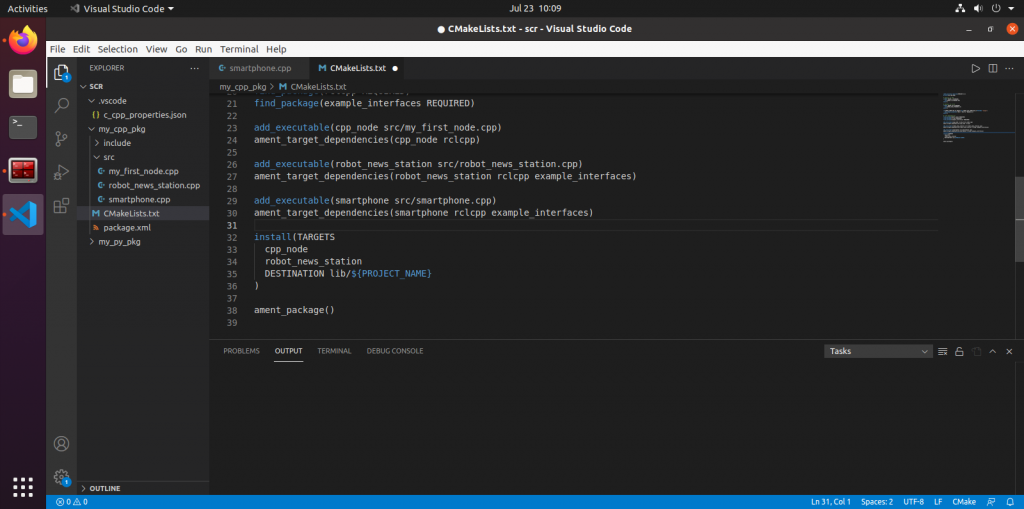
}



เปิดไฟล์ CMakeLists.txt เพิ่มโค้ดบรรทัดที่ 29 และ 30 ด้วยโค้ดด้านล่าง

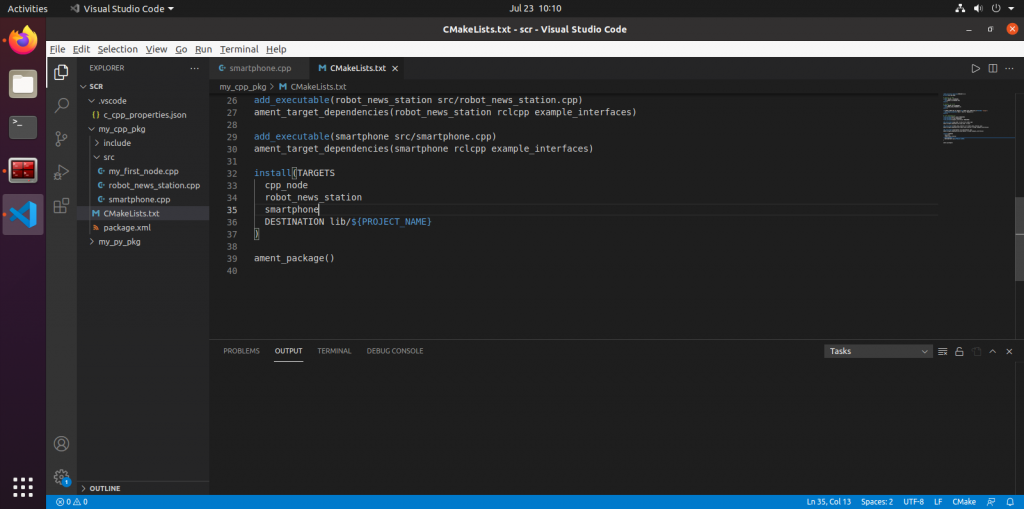
add\_executable(smartphone src/smartphone.cpp)

ament\_target\_dependencies(smartphone rclcpp example\_interfaces)



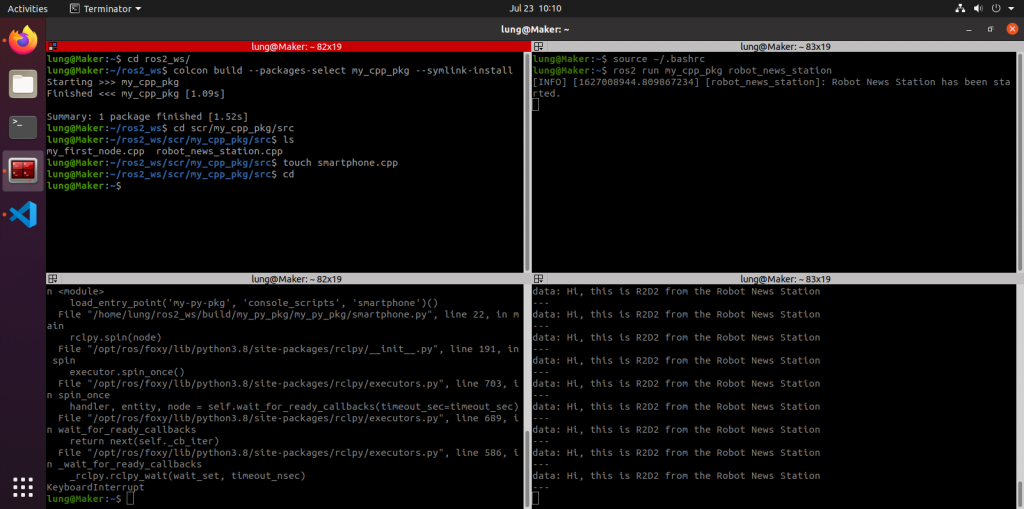
เพิ่มโค้ดบรรทัดที่ 35 ด้วยโค้ดด้านล่าง

smartphone

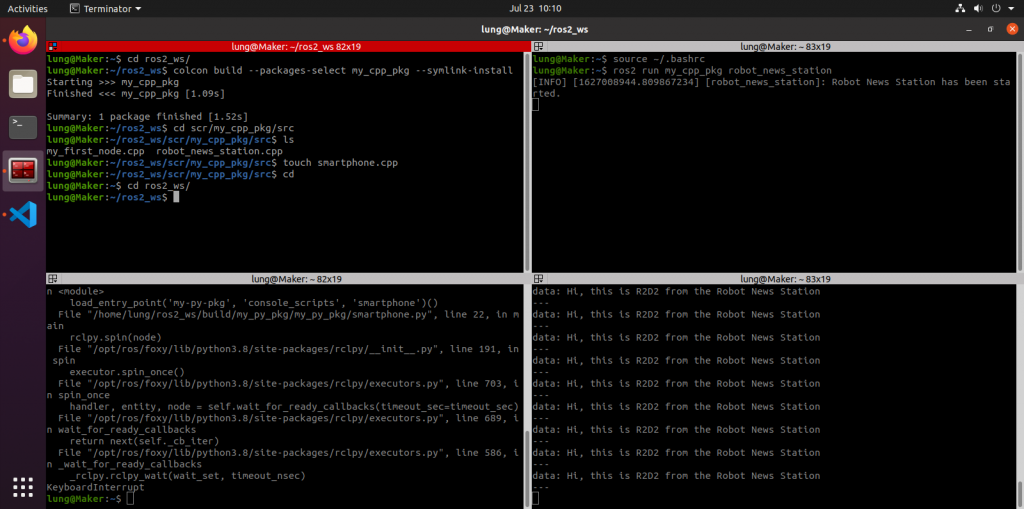


ไปที่ หน้าต่างที่ 1 กลับสู่ Home เพื่อจะ Build แพ็กเกจ my\_cpp\_pkg

cd

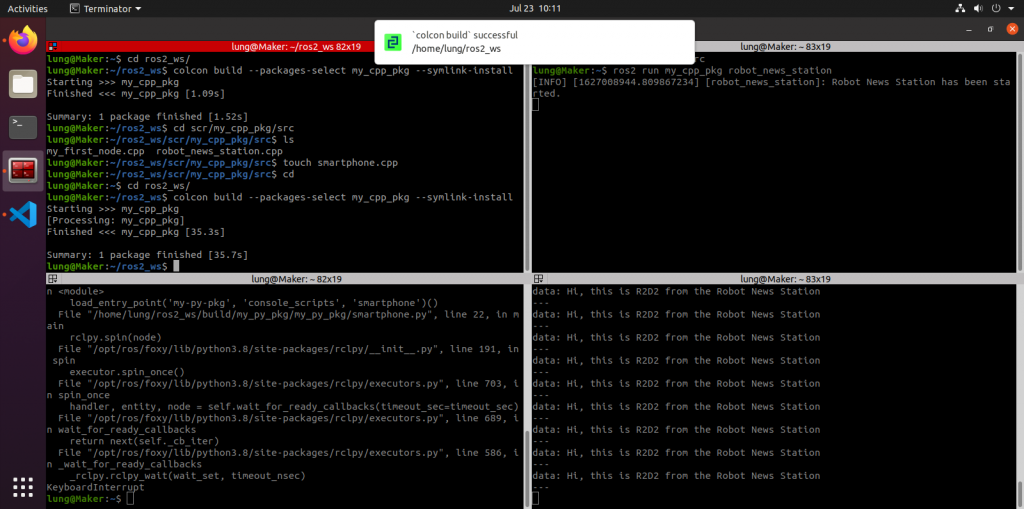


cd ros2\_ws/



คำสั่ง Build เฉพาะ แพ็กเกจ my\_cpp\_pkg

colcon build --packages-select my\_cpp\_pkg --symlink-install



ไปที่ หน้าต่างที่ 3 โดยเราจะต้องทําให้ Terminator รู้จักคําสั่งของ ROS2 ด้วย คําสั่ง

source ~/.bashrc



เรียกใช้งาน smartphone ด้วย ROS2 จะแสดงการรับ Message จากที่ส่งออกมาจาก topic robot\_news ภาษา C++

ros2 run my\_cpp\_pkg smartphone

