THE HONG KONG POLYTECHNIC UNIVERSITY
香港理工大學

Department of Aeronautical and Aviation Engineering
航空及民航工程學系

Electronics & Information Technologies for Unmanned Aerial Systems

Week 3: Project Session 1

Dr. Hailong Huang
Assistant Professor
Department of Aeronautical and Aviation Engineering
The Hong Kong Polytechnic University

Outline

1. ROS2 Project Creation

2. An Example of using Topics Communication

3. Project Assignment 1

Prepared Works

- **Connect your computer with the monitor, keyboard, and mouse**
- **Turn on your computer**
  password: 1234
- **Download project resources**

```
# download resources from Github
git clone https://github.com/AIMSPolyU/AAE4202_project.git
```

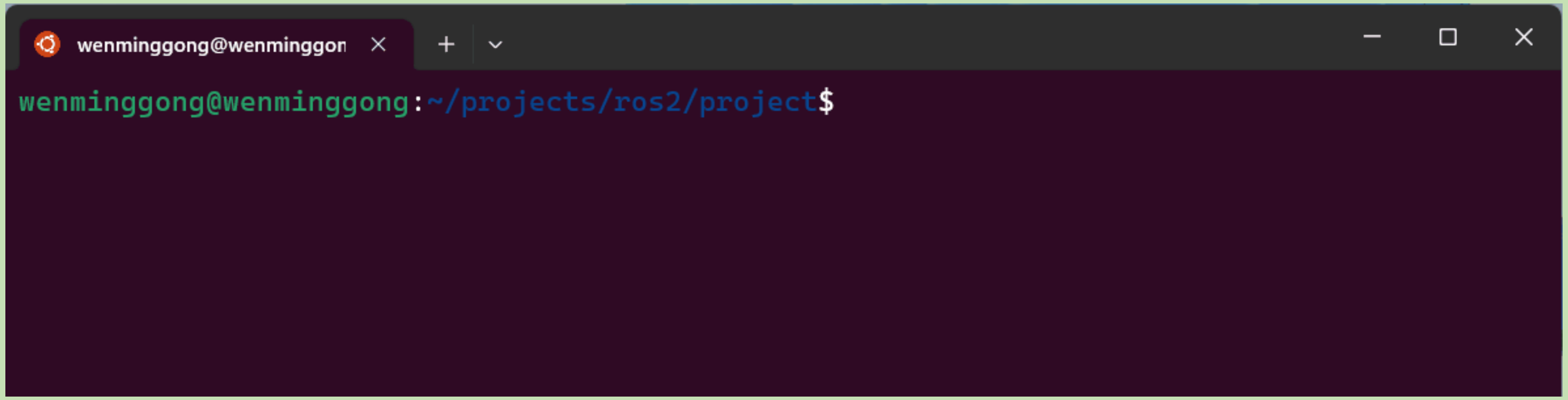
Outline

1. ROS2 Project Creation
2. An Example of using Topics Communication
3. Project Assignment 1

1. ROS2 Project Creation

Create a new ROS2 project from scratch

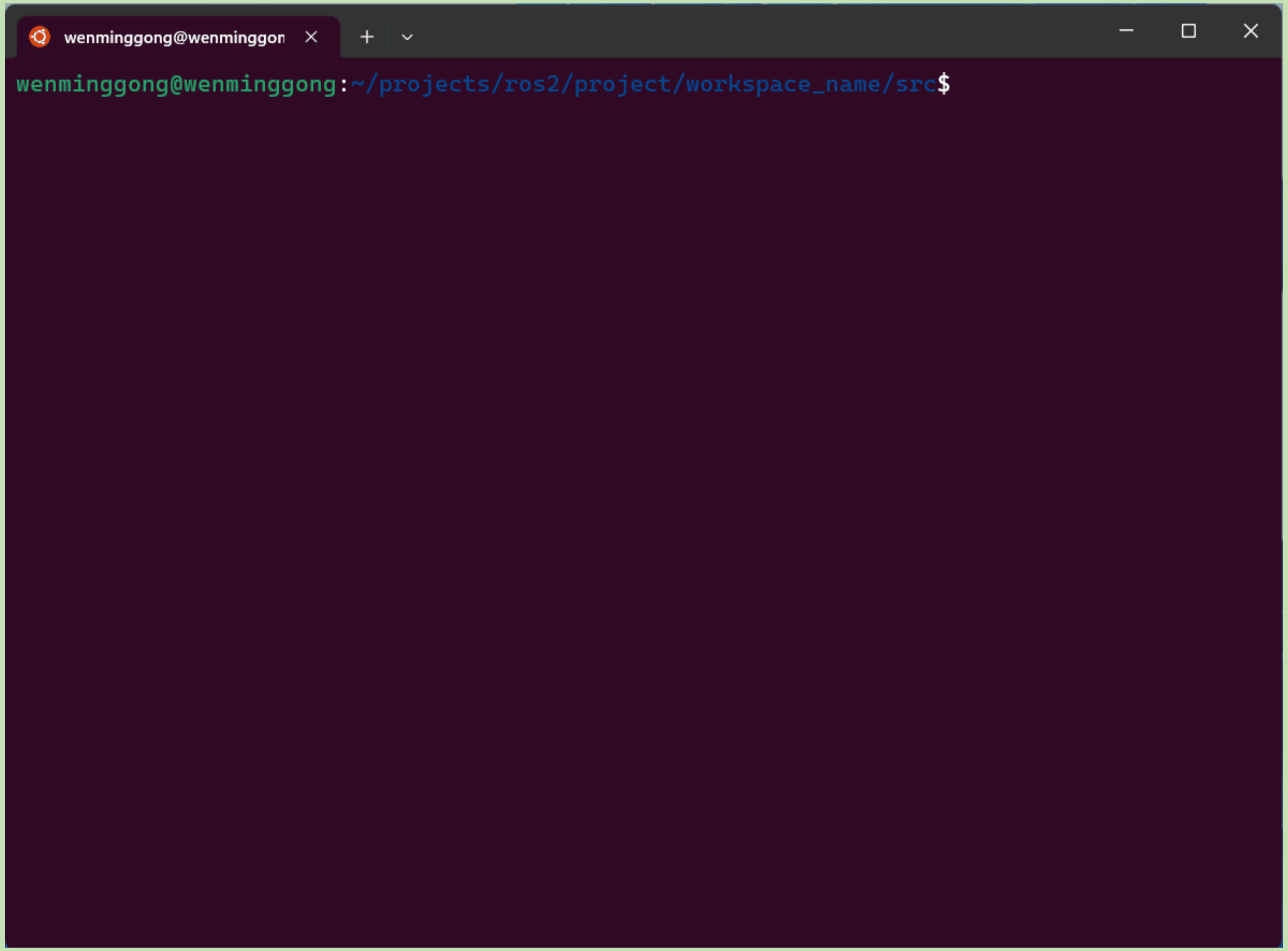
```
# 1. create a workspace
mkdir –p workspace_name/src

# 2. enter workspace_name/src
cd workspace_name/src
```



```
wenminggong@wenminggong:~/projects/ros2/project$
```

1. ROS2 Project Creation

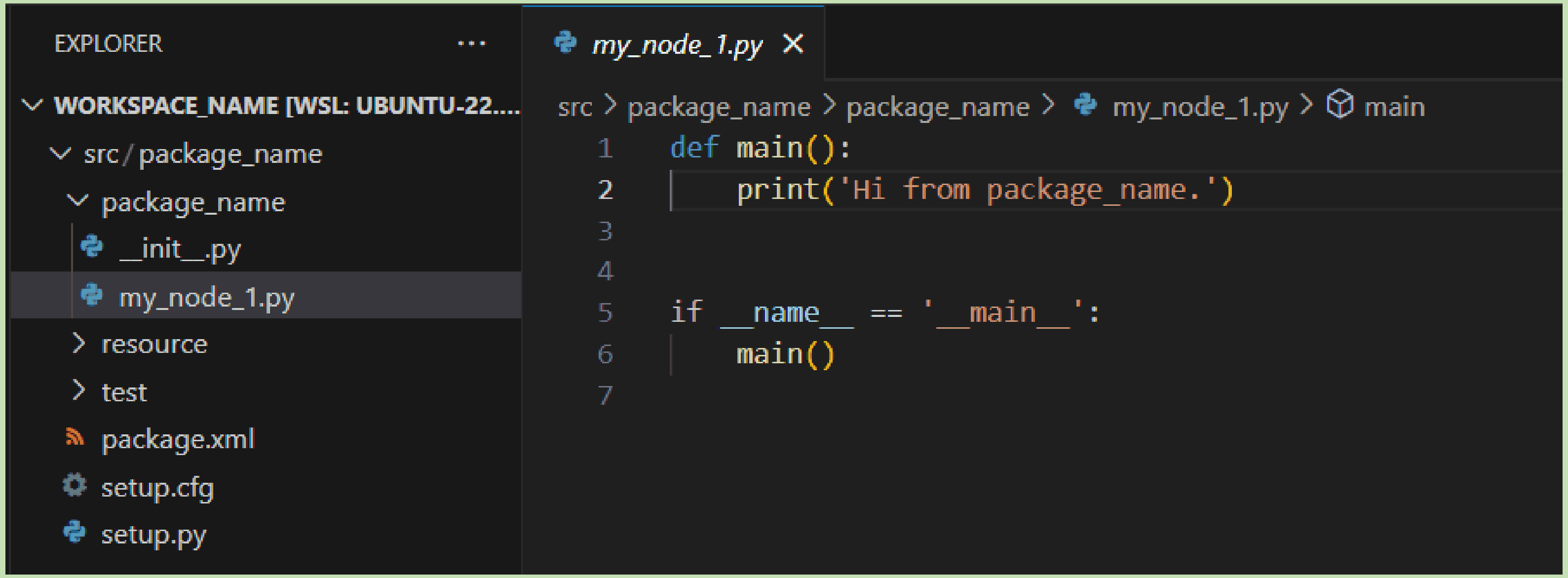
# 3. create a new python package with a node
ros2 pkg create package_name --build-type ament_python --dependencies rclpy --node-name my_node_1



```
wenminggong@wenminggong:~/projects/ros2/project/workspace_name/src$
```

1. ROS2 Project Creation

# 4. open workspace_name/src/package_name/package_name/my_node_1.py and write your code



EXPLORER

```
WORKSPACE_NAME [WSL: UBUNTU-22....
  src / package_name
    package_name
      __init__.py
      my_node_1.py
    resource
    test
    package.xml
    setup.cfg
    setup.py
```

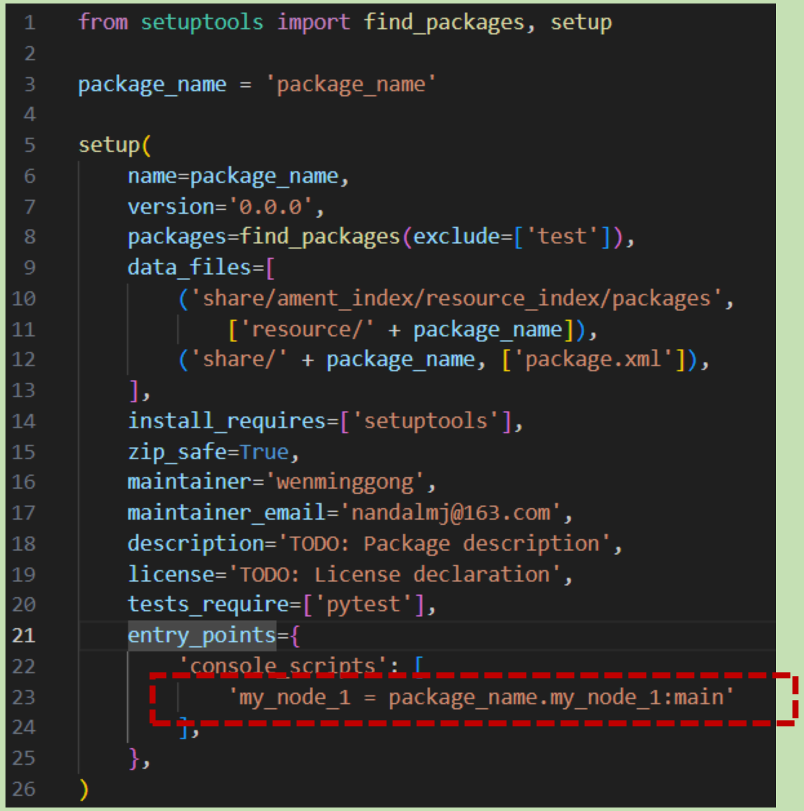
my_node_1.py

src > package_name > package_name > my_node_1.py > main

```python
def main():
    print('Hi from package_name.')


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

1. ROS2 Project Creation

# 5. modify configuration file: add node projection in workspace_name/src/package_name/setup.py

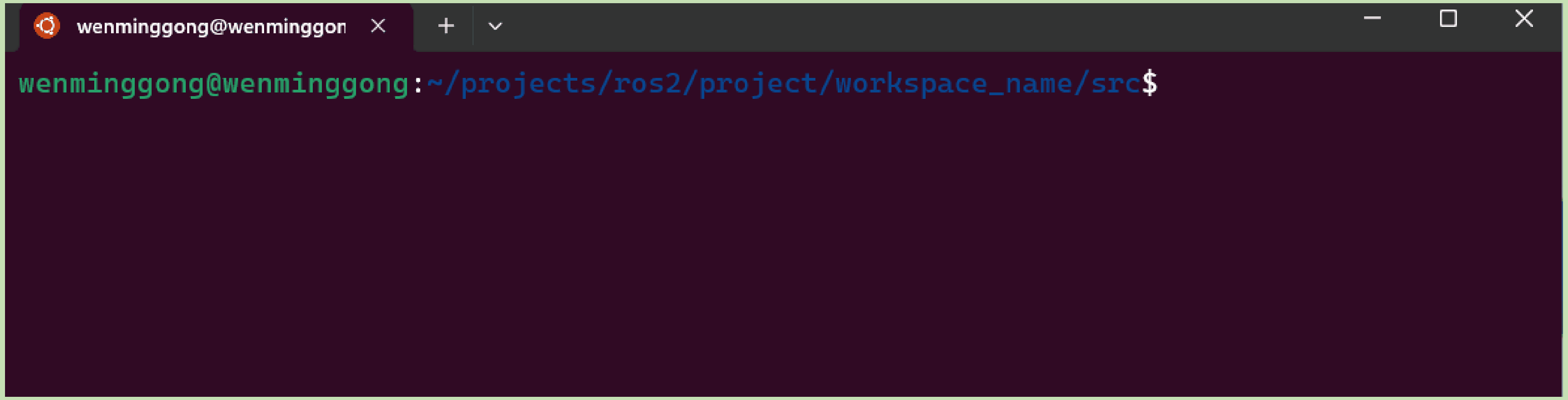


```
from setuptools import find_packages, setup

package_name = 'package_name'

setup(
    name=package_name,
    version='0.0.0',
    packages=find_packages(exclude=['test']),
    data_files=[
        ('share/ament_index/resource_index/packages',
            ['resource/' + package_name]),
        ('share/' + package_name, ['package.xml']),
    ],
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='wenminggong',
    maintainer_email='nandalmj@163.com',
    description='TODO: Package description',
    license='TODO: License declaration',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
            'my_node_1 = package_name.my_node_1:main'
        ],
    },
)
```

# add for every node, if a node is created using the command "*ros2 pkg create …", it* will be added by default

1. ROS2 Project Creation

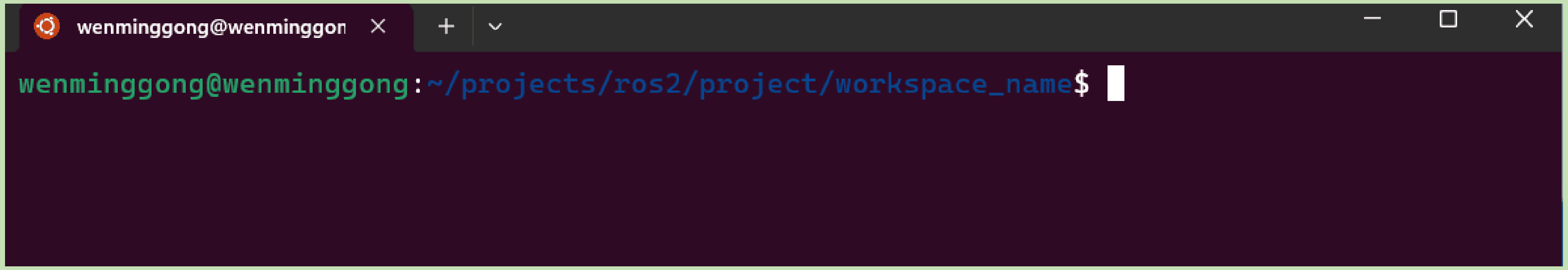
```
# 6. build and compile project
# 6.1 go back to workspace_name directory
cd ..

# 6.2 use colcon tool to build and compile project
colcon build --packages-select package_name
```



```
wenminggong@wenminggong:~/projects/ros2/project/workspace_name/src$
```

1. ROS2 Project Creation

```
# 7. set environment variables for ROS2
. install/setup.bash

# 8. run my_node_1
ros2 run package_name my_node_1
```



```
wenminggong@wenminggong:~/projects/ros2/project/workspace_name$
```

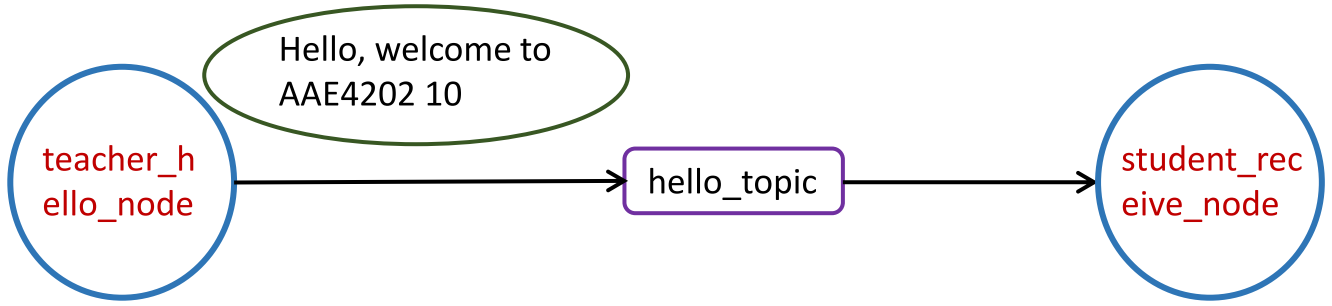
Outline

1. ROS2 Project Creation
2. An Example of using Topics Communication
3. Project Assignment 1

2. An Example of using Topics Communication

Task Description

- Practice using **topics** communication as it is most frequently used
- A teacher node called *teacher_hello_node,* it is responsible for publishing a "hello topic" to the student node
  - ➢ Send the hello message at 1hz (once a second)
  - ➢ Hello message with increased number: "Hello, welcome to AAE4202 10"
- A student node called *student_receive_node*, it is required to receive hello message from the teacher node
  - ➢ Print the message in the screen



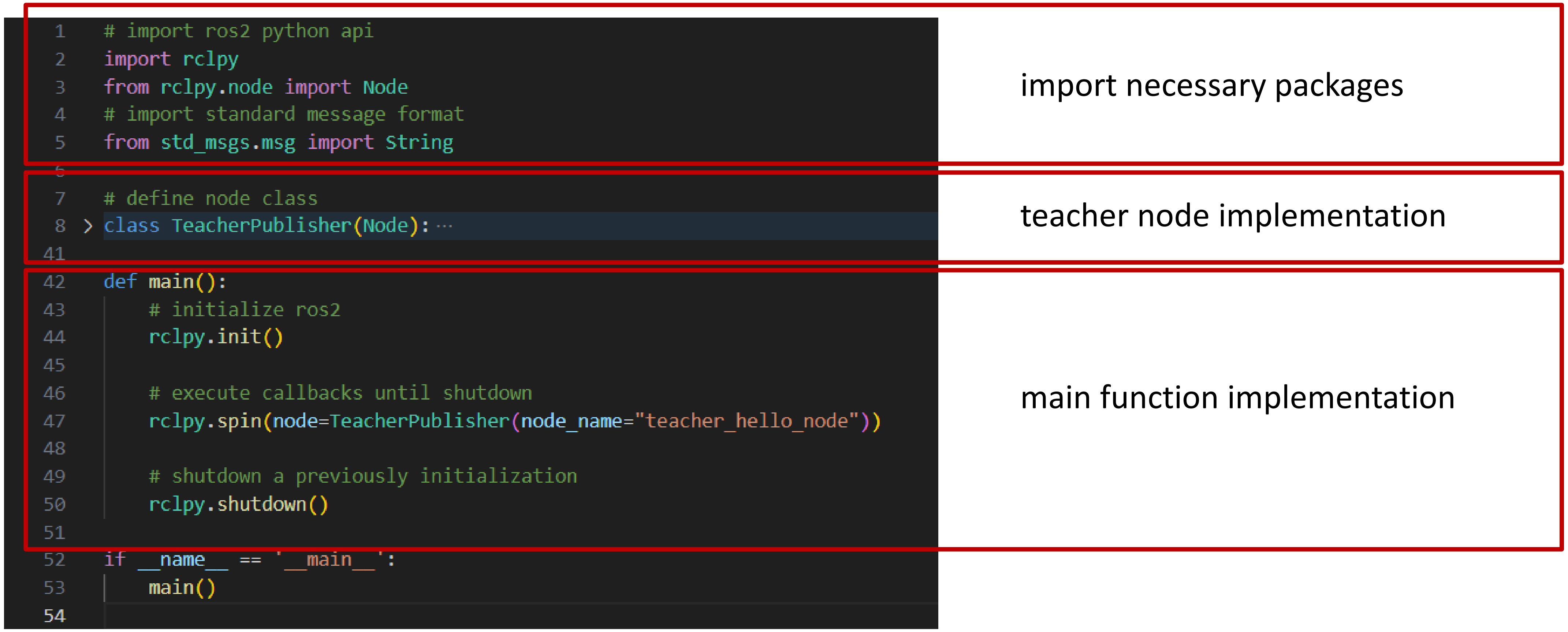
2. An Example of using Topics Communication

Procedures

- Create new project (workspace, package, node)
- Write down codes for publisher node and subscriber node
- Modify configuration file
- Compile and run

2. An Example of using Topics Communication

teacher_hello_node



```
# import ros2 python api
import rclpy
from rclpy.node import Node
# import standard message format
from std_msgs.msg import String

# define node class
class TeacherPublisher(Node): ···

def main():
    # initialize ros2
    rclpy.init()

    # execute callbacks until shutdown
    rclpy.spin(node=TeacherPublisher(node_name="teacher_hello_node"))

    # shutdown a previously initialization
    rclpy.shutdown()

if __name__ == '__main__':
    main()

```

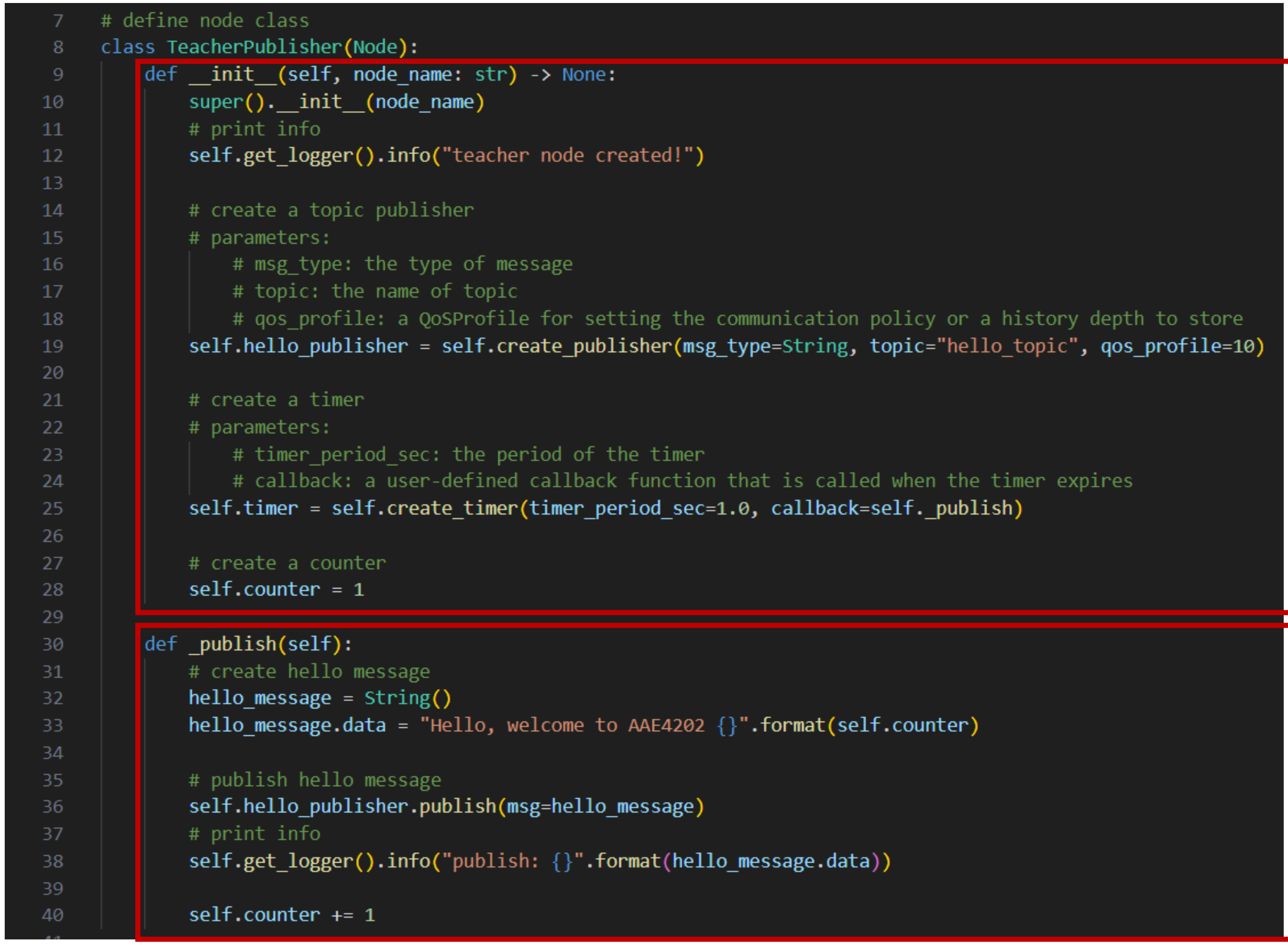
import necessary packages

teacher node implementation

main function implementation

2. An Example of using Topics Communication

teacher_hello_node



```python
# define node class
class TeacherPublisher(Node):
    def __init__(self, node_name: str) -> None:
        super().__init__(node_name)
        # print info
        self.get_logger().info("teacher node created!")

        # create a topic publisher
        # parameters:
            # msg_type: the type of message
            # topic: the name of topic
            # qos_profile: a QoSProfile for setting the communication policy or a history depth to store
        self.hello_publisher = self.create_publisher(msg_type=String, topic="hello_topic", qos_profile=10)

        # create a timer
        # parameters:
            # timer_period_sec: the period of the timer
            # callback: a user-defined callback function that is called when the timer expires
        self.timer = self.create_timer(timer_period_sec=1.0, callback=self._publish)

        # create a counter
        self.counter = 1

    def _publish(self):
        # create hello message
        hello_message = String()
        hello_message.data = "Hello, welcome to AAE4202 {}".format(self.counter)

        # publish hello message
        self.hello_publisher.publish(msg=hello_message)
        # print info
        self.get_logger().info("publish: {}".format(hello_message.data))

        self.counter += 1
```

initialization function

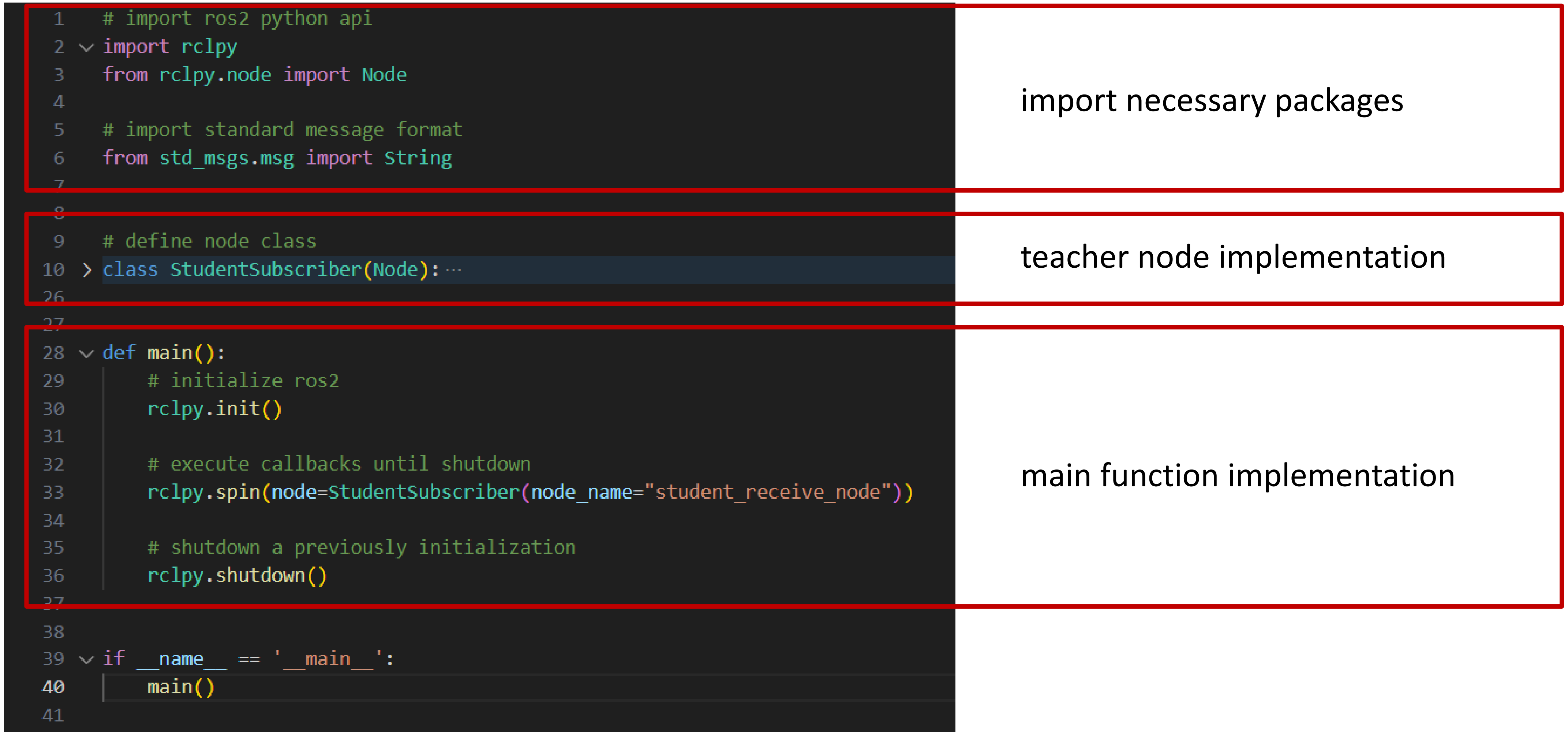
- print log info
- create a topic publisher
- create a timer

publishing function

- create hello message
- publish hello message
- print log info

2. An Example of using Topics Communication

student_receive_node



```python
# import ros2 python api
import rclpy
from rclpy.node import Node

# import standard message format
from std_msgs.msg import String


# define node class
class StudentSubscriber(Node):···


def main():
    # initialize ros2
    rclpy.init()

    # execute callbacks until shutdown
    rclpy.spin(node=StudentSubscriber(node_name="student_receive_node"))

    # shutdown a previously initialization
    rclpy.shutdown()


if __name__ == '__main__':
    main()

```

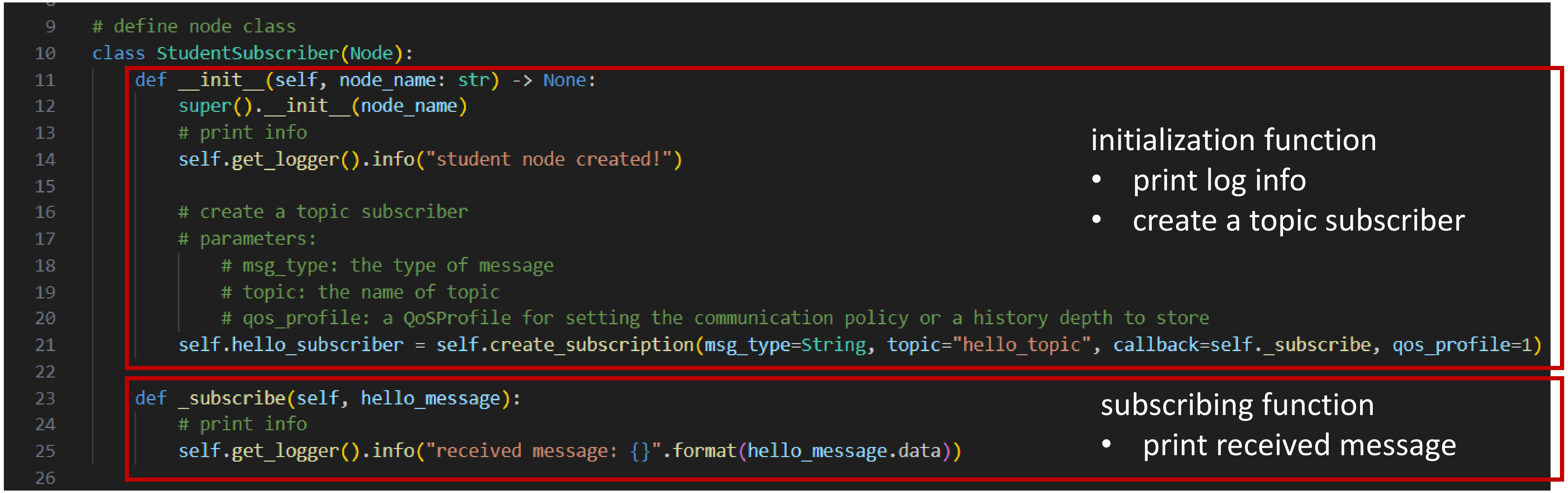
import necessary packages

teacher node implementation

main function implementation

2. An Example of using Topics Communication

student_receive_node



```python
# define node class
class StudentSubscriber(Node):
    def __init__(self, node_name: str) -> None:
        super().__init__(node_name)
        # print info
        self.get_logger().info("student node created!")

        # create a topic subscriber
        # parameters:
            # msg_type: the type of message
            # topic: the name of topic
            # qos_profile: a QoSProfile for setting the communication policy or a history depth to store
        self.hello_subscriber = self.create_subscription(msg_type=String, topic="hello_topic", callback=self._subscribe, qos_profile=1)

    def _subscribe(self, hello_message):
        # print info
        self.get_logger().info("received message: {}".format(hello_message.data))
```

initialization function

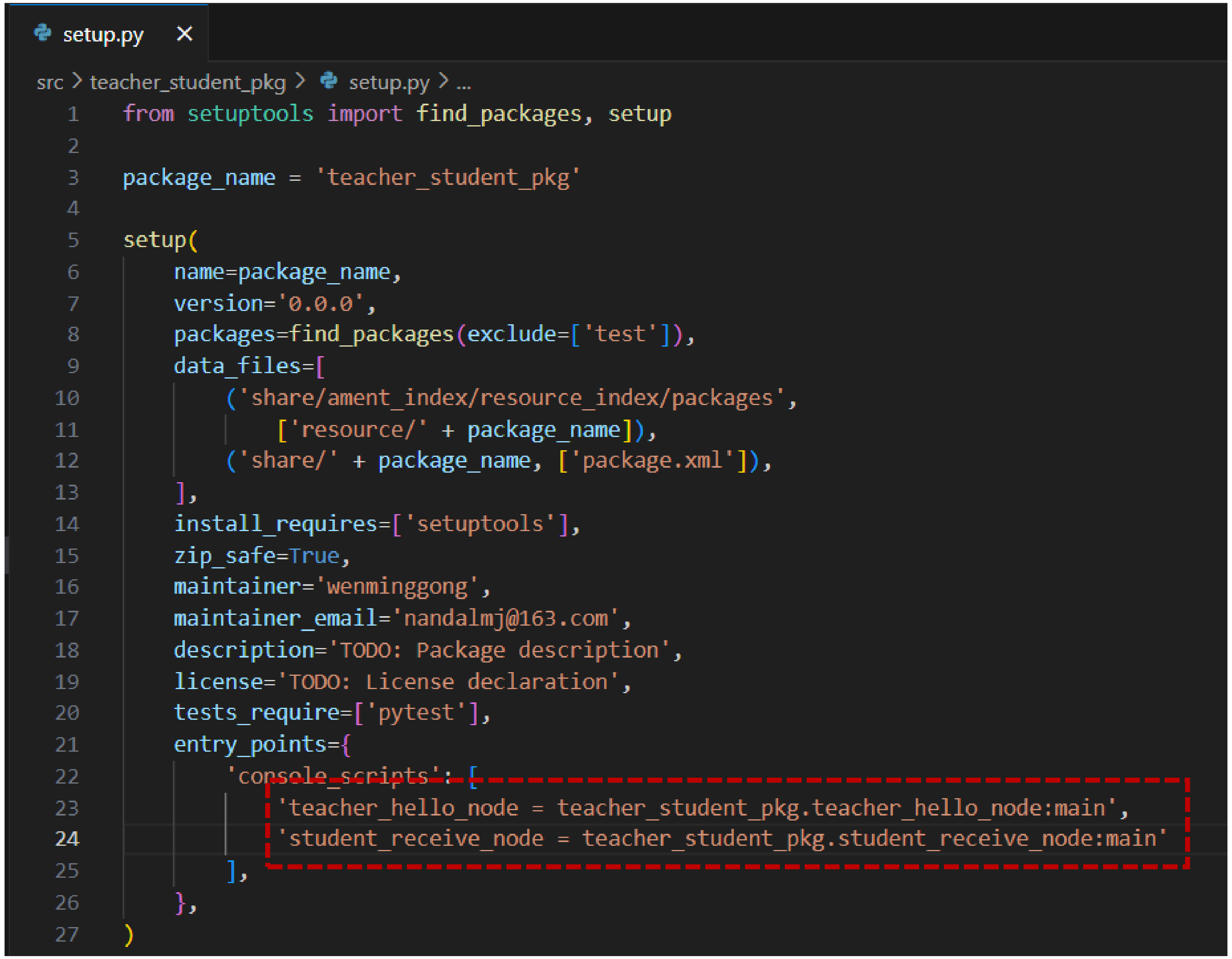
- print log info
- create a topic subscriber

subscribing function

- print received message

2. An Example of using Topics Communication

modify configuration file: setup.py

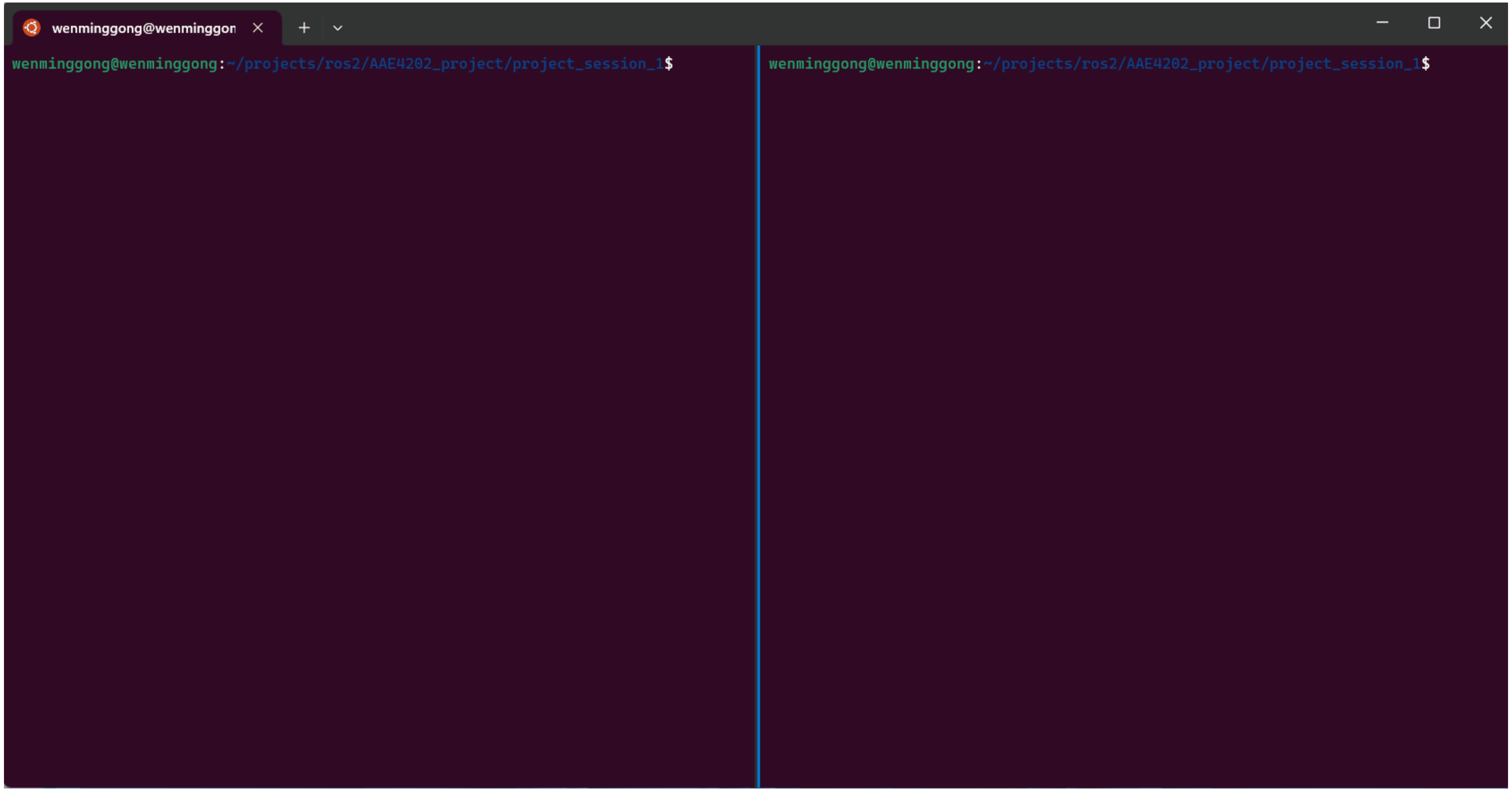


```python
from setuptools import find_packages, setup

package_name = 'teacher_student_pkg'

setup(
    name=package_name,
    version='0.0.0',
    packages=find_packages(exclude=['test']),
    data_files=[
        ('share/ament_index/resource_index/packages',
            ['resource/' + package_name]),
        ('share/' + package_name, ['package.xml']),
    ],
    install_requires=['setuptools'],
    zip_safe=True,
    maintainer='wenminggong',
    maintainer_email='nandalmj@163.com',
    description='TODO: Package description',
    license='TODO: License declaration',
    tests_require=['pytest'],
    entry_points={
        'console_scripts': [
            'teacher_hello_node = teacher_student_pkg.teacher_hello_node:main',
            'student_receive_node = teacher_student_pkg.student_receive_node:main'
        ],
    },
)
```

# add node projection

2. An Example of using Topics Communication

Experimental Results



```
wenminggong@wenminggong:~/projects/ros2/AAE4202_project/project_session_1$
```

```
wenminggong@wenminggong:~/projects/ros2/AAE4202_project/project_session_1$
```

Outline

1. ROS2 Project Creation
2. An Example of using Topics Communication
3. Project Assignment 1

3. Project Assignment 1

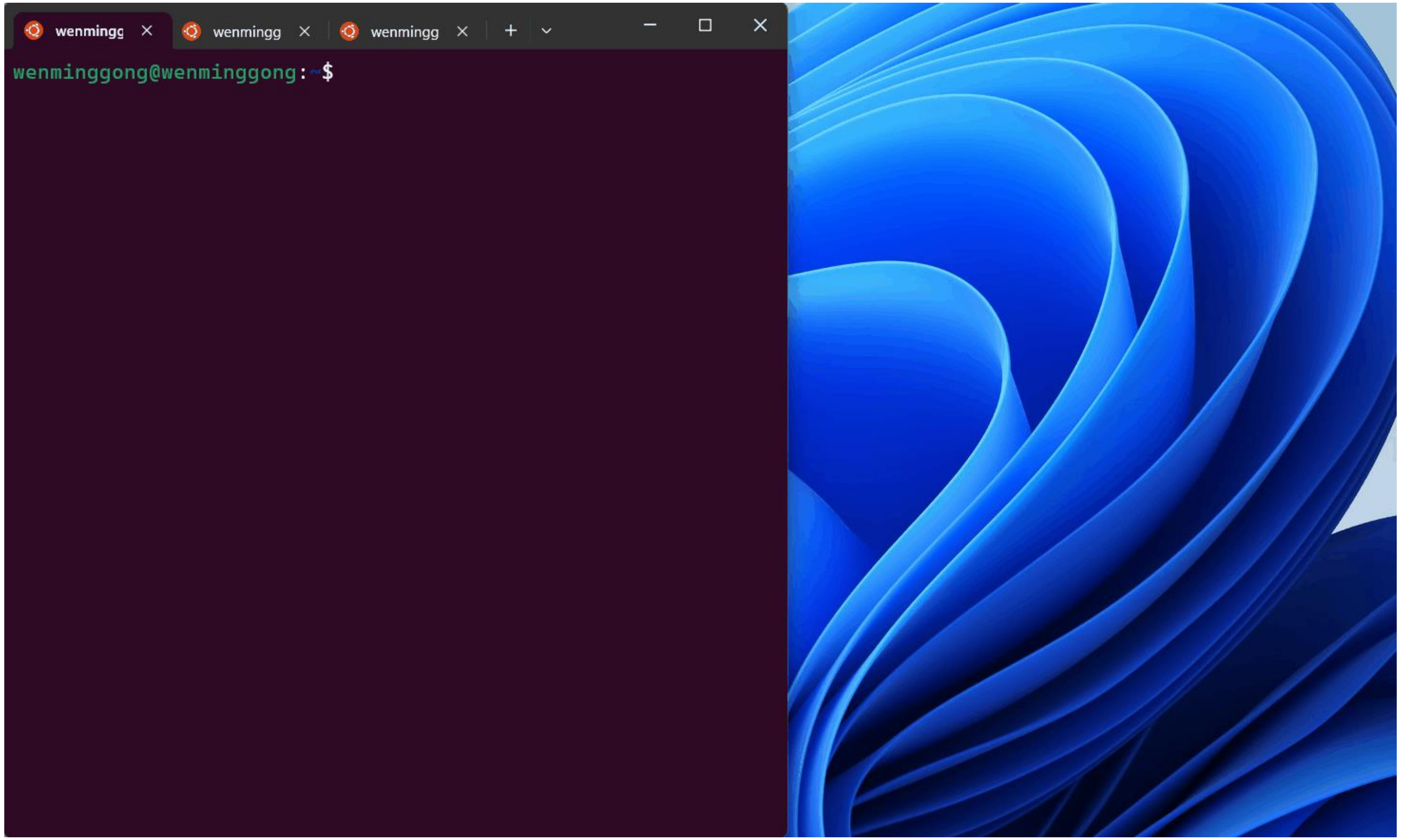
Prerequisites

- Turtlesim: a lightweight simulator for learning ROS2, it illustrates what ROS2 does at the most basic level

- Two common nodes
  - ➢ turtlesim_node: open the turtlesim simulator
  - ➢ turtle_teleop_key: control the turtle in the simulator by keyboard

3. Project Assignment 1

Prerequisites



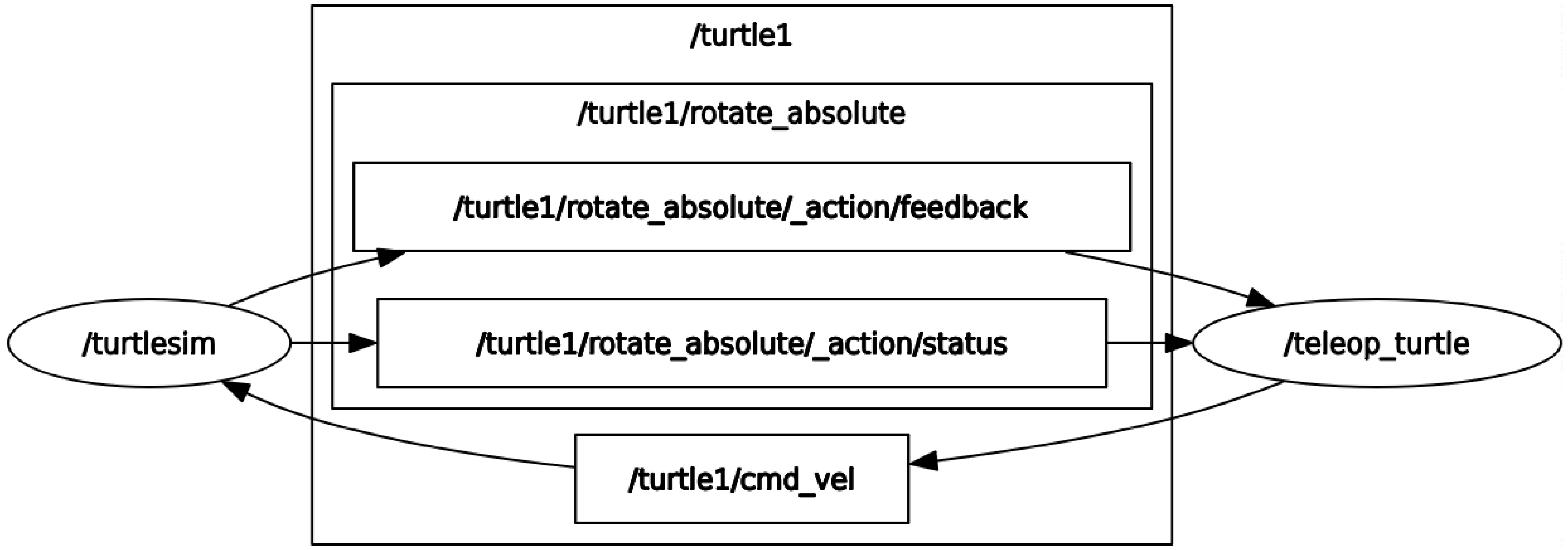
```
wenminggong@wenminggong:~$
```

3. Project Assignment 1

Prerequisites

- rqt: a graphical user interface (GUI) tool for ROS2
- Use rqt tool to show the ROS2 graph

```
rqt_graph
```


/turtle1
/turtle1/rotate_absolute
/turtle1/rotate_absolute/_action/feedback
/turtlesim
/turtle1/rotate_absolute/_action/status
/teleop_turtle
/turtle1/cmd_vel

3. Project Assignment 1

Task Description

- Implement a turtle control node
  - ➢ Control the turtle to move around a circle
  - ➢ Publish control message at a frequency of 2hz
  - ➢ The radius of the circle is 2

- Implement a turtle pose show node
  - ➢ Print the turtle's position in real time, including x, y, and theta
  - ➢ Print the radius of the circle which the turtle follows

3. Project Assignment 1

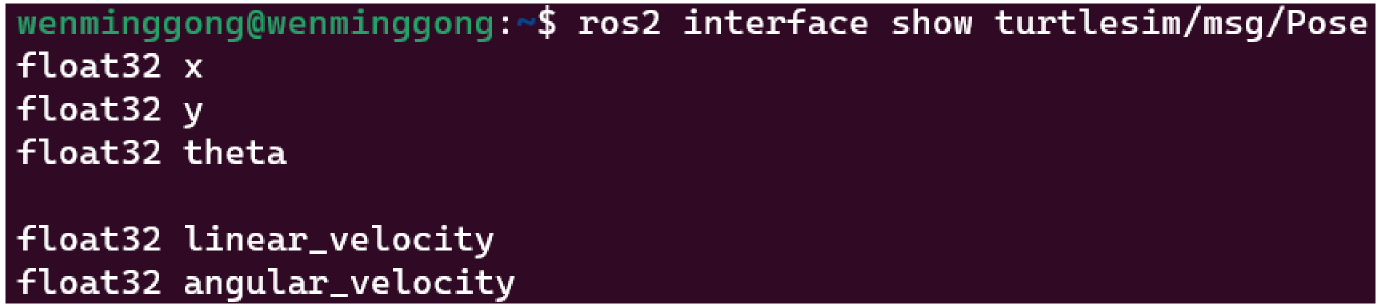
Task Description

- Assignment requirements
  - Record the process and the results of this experiment
  - Write the experiment report to describe the experiment process and results
  - Send the experiment report to the TA (Liu) by email before the next project session (before Thursday week 5)
  - Only one submission per group
  - Please submit the report in pdf format and sign the names of all members

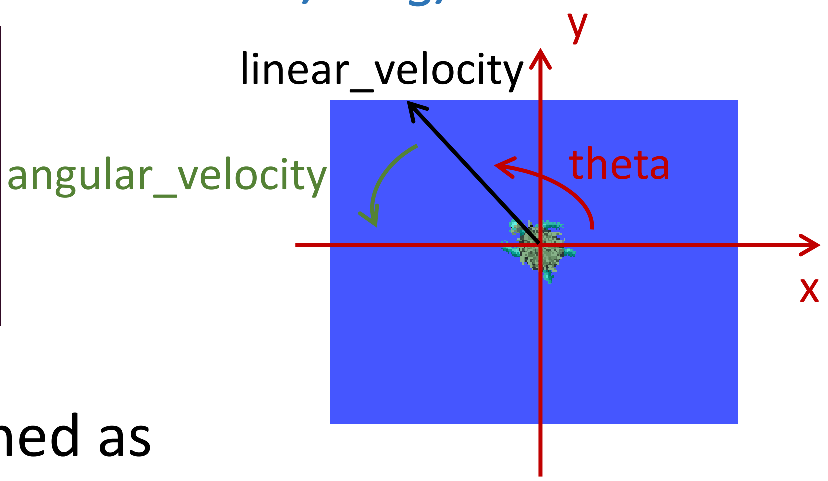
3. Project Assignment 1

Task Description

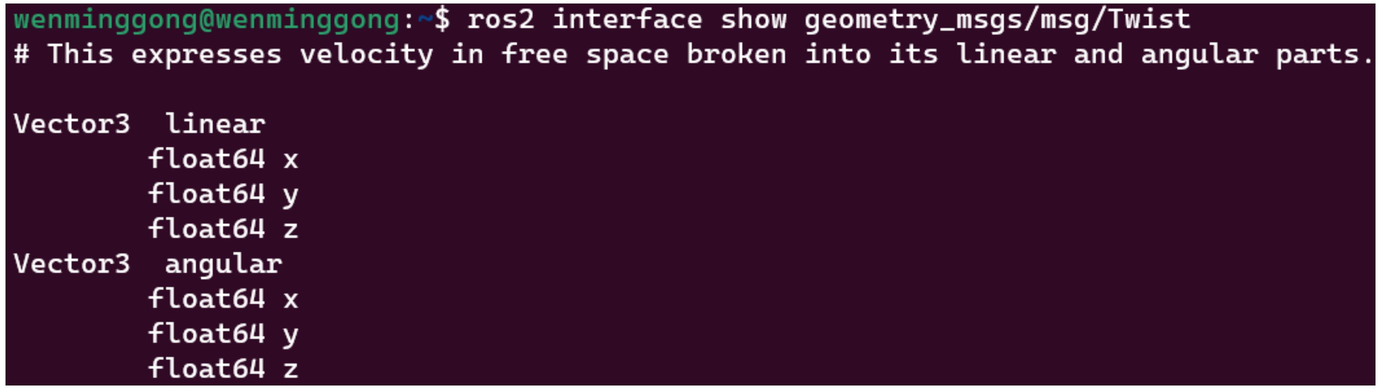
- The turtlesim_node publishes its pose through */turtle1/pose* topic, and subscribes control command through */turtle1/cmd_vel* topic
- The message format of */turtle1/pose* topic is defined as *turtlesim/msg/Pose*



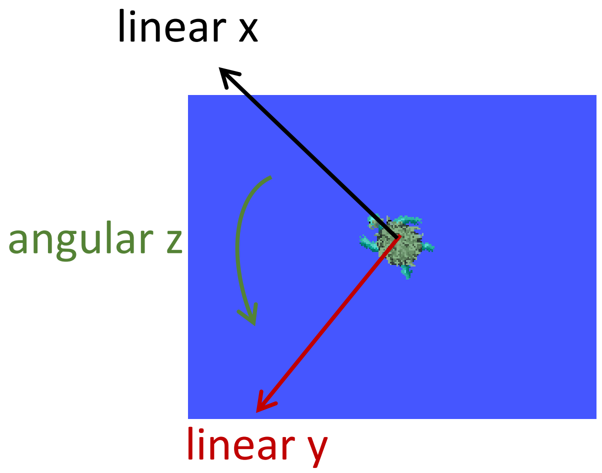
```
wenminggong@wenminggong:~$ ros2 interface show turtlesim/msg/Pose
float32 x
float32 y
float32 theta

float32 linear_velocity
float32 angular_velocity
```



- The message format of */turtle1/cmd_vel* topic is defined as *geometry_msgs/msg/Twist*



```
wenminggong@wenminggong:~$ ros2 interface show geometry_msgs/msg/Twist
# This expresses velocity in free space broken into its linear and angular parts.

Vector3  linear
        float64 x
        float64 y
        float64 z
Vector3  angular
        float64 x
        float64 y
        float64 z
```



3. Project Assignment 1

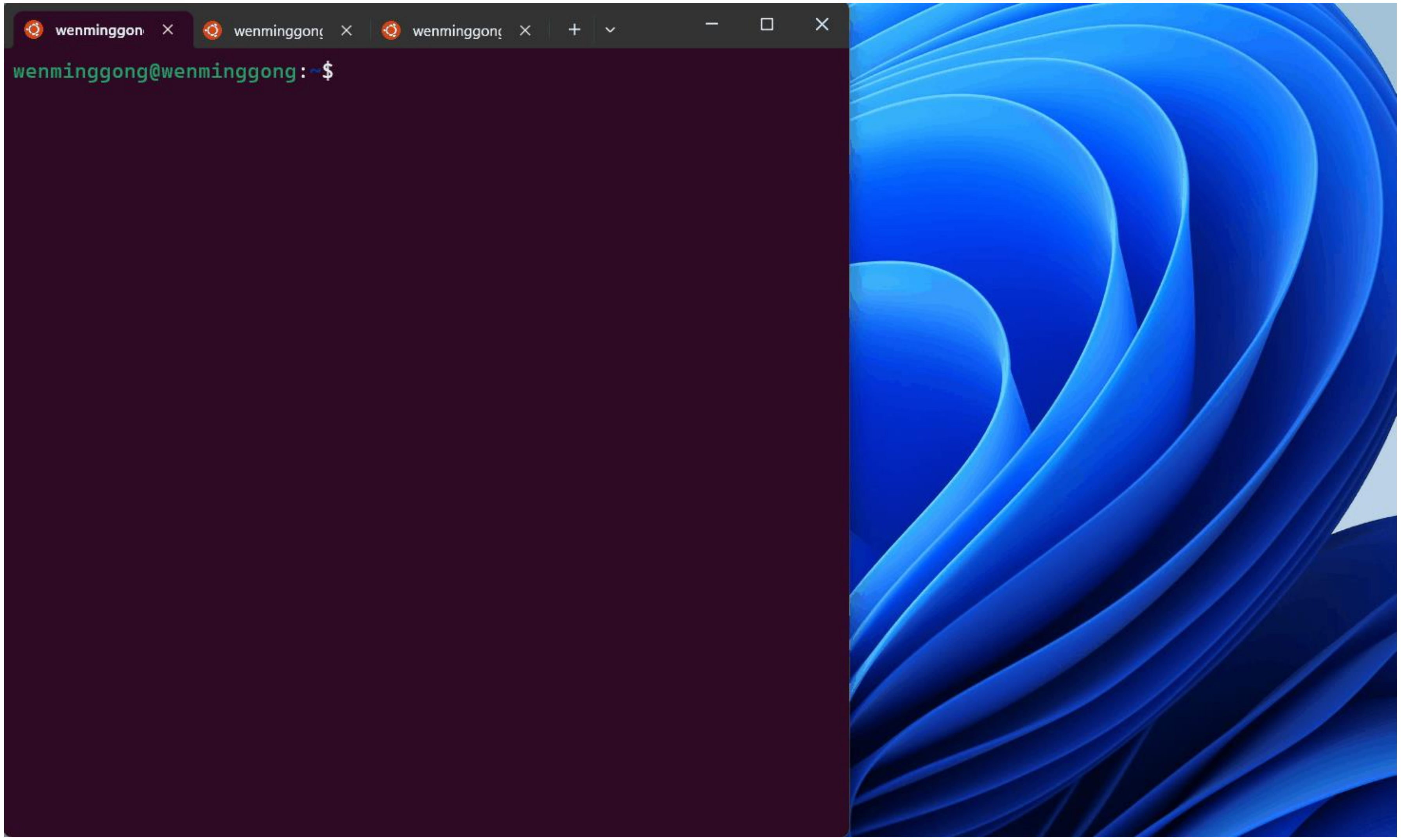
Task Description

- Message dependencies need to be added when the package is created

```
# create a new python package with a node
ros2 pkg create package_name --build-type ament_python --dependencies rclpy
turtlesim geometry_msgs --node-name my_node_1
```

3. Project Assignment 1

Demonstration



```
wenminggong@wenminggong:~$
```

Thank You!

Dr. Hailong Huang

Assistant Professor

Department of Aeronautical and Aviation Engineering

The Hong Kong Polytechnic University

https://sites.google.com/view/hailong-huang/home

Email: hailong.huang@polyu.edu.hk