



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

Slide 1 of 28

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

- **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
```

1. ROS2 Project Creation
2. Task Example
3. Project Assignment

1. ROS2 Project Creation

2. Task Example

3. Project Assignment

1. ROS2 Project Creation

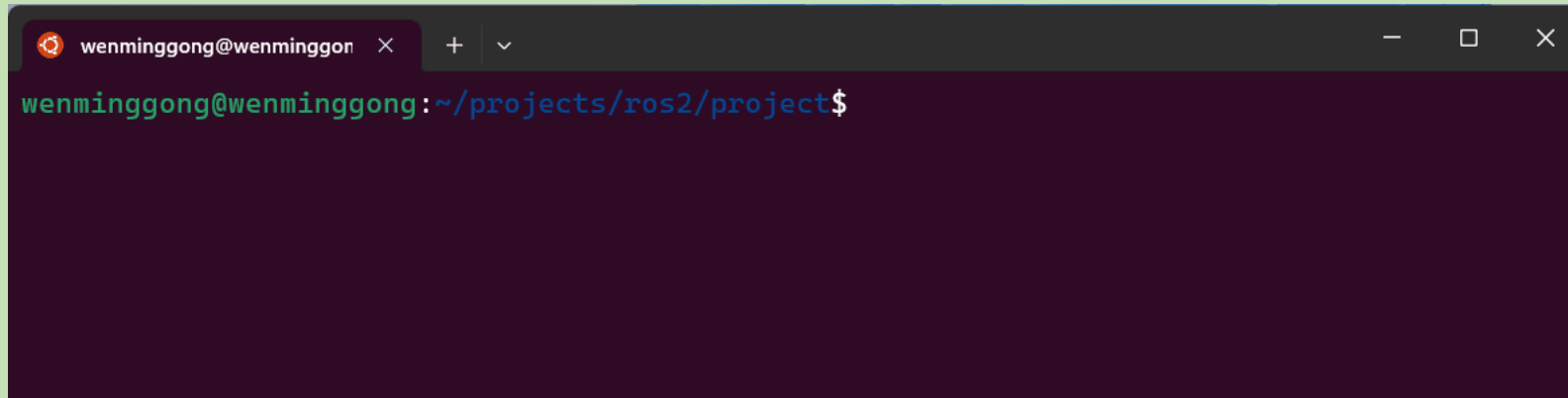
Create a new ROS2 project from scratch

1. create a workspace

mkdir -p workspace_name/src

2. enter workspace_name/src

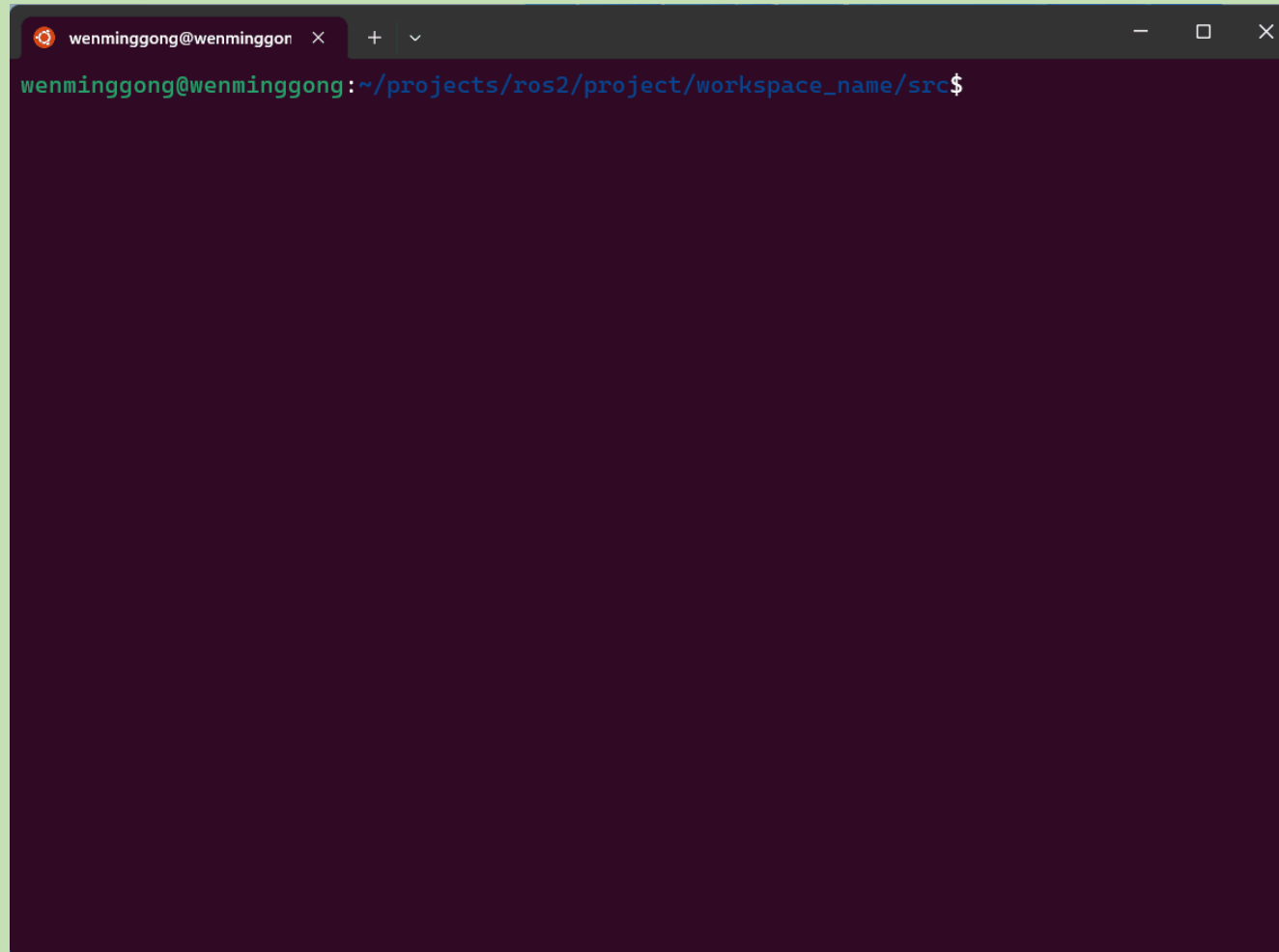
cd workspace_name/src

A terminal window with a dark background and light text. The window title bar shows 'wenminggong@wenminggon' and standard window controls. The terminal text shows the user 'wenminggong' at the prompt 'wenminggong@wenminggong' with the current directory path '~ /projects/ros2/project\$'.

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

1. ROS2 Project Creation

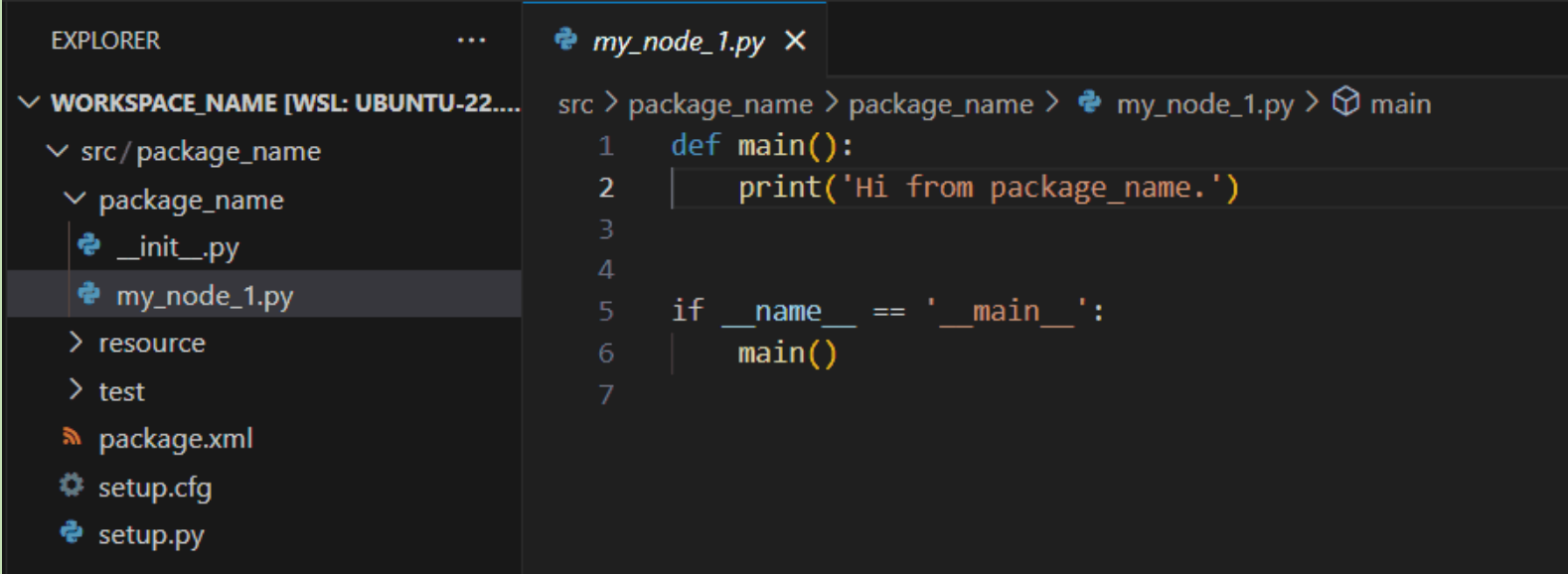
3. create a new python package with a node
ros2 pkg create package_name --build-type ament_python --dependencies rclpy
--node-name my_node_1

A terminal window with a dark purple background and a light green title bar. The title bar contains the text 'wenminggong@wenminggon' followed by a close button 'X' and window control buttons '+', 'v', '-', '□', and 'X'. The terminal shows the command 'ros2 pkg create package_name --build-type ament_python --dependencies rclpy --node-name my_node_1' being entered at the prompt 'wenminggong@wenminggong: ~/projects/ros2/project/workspace_name/src\$'.

```
wenminggong@wenminggong: ~/projects/ros2/project/workspace_name/src$ ros2 pkg create package_name --build-type ament_python --dependencies rclpy --node-name my_node_1
```

1. ROS2 Project Creation

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



The screenshot shows the Visual Studio Code interface. On the left, the 'EXPLORER' sidebar displays the project structure. The workspace is named 'WORKSPACE_NAME [WSL: UBUNTU-22...]' and contains a folder 'src/package_name'. Inside this folder is a sub-folder 'package_name' which contains files '__init__.py', 'my_node_1.py', 'resource', 'test', 'package.xml', 'setup.cfg', and 'setup.py'. The file 'my_node_1.py' is selected. The main editor window shows the code for 'my_node_1.py' with the following content:

```
src > package_name > package_name > my_node_1.py > main
1  def main():
2      print('Hi from package_name.')
3
4
5  if __name__ == '__main__':
6      main()
7
```

1. ROS2 Project Creation

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

```
1  from setuptools import find_packages, setup
2
3  package_name = 'package_name'
4
5  setup(
6      name=package_name,
7      version='0.0.0',
8      packages=find_packages(exclude=['test']),
9      data_files=[
10         ('share/ament_index/resource_index/packages',
11          ['resource/' + package_name]),
12         ('share/' + package_name, ['package.xml']),
13     ],
14     install_requires=['setuptools'],
15     zip_safe=True,
16     maintainer='wenminggong',
17     maintainer_email='nandalmj@163.com',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23             'my_node_1 = package_name.my_node_1:main'
24         ],
25     },
26 )
```

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

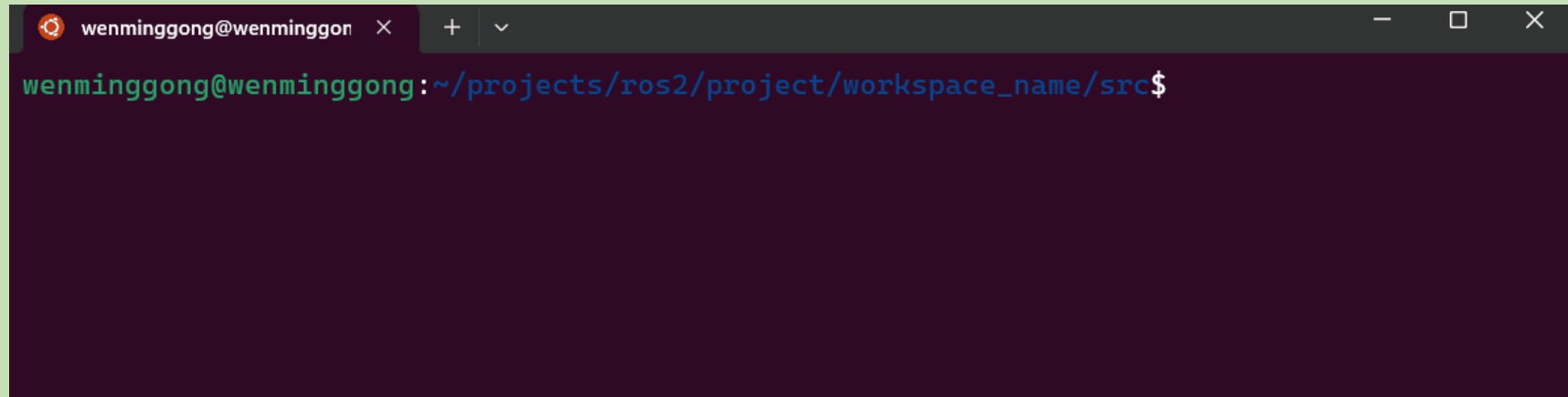
6. build and compile project

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

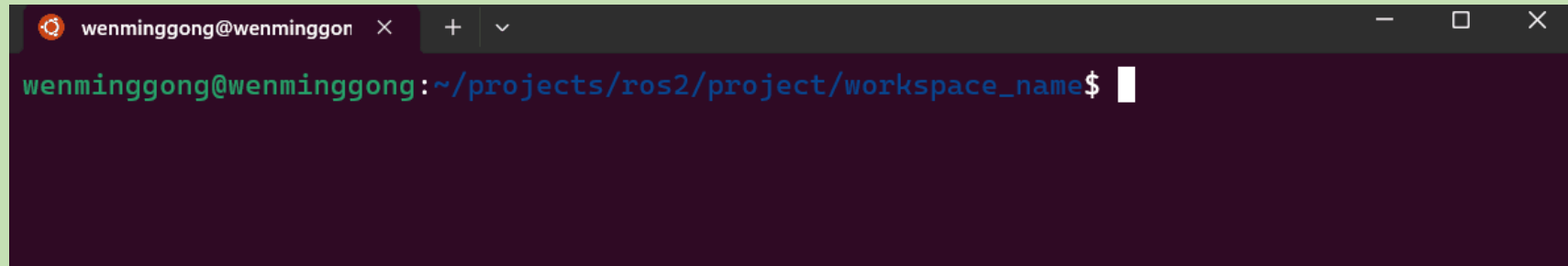
A terminal window with a dark purple background and a light green title bar. The title bar contains the text 'wenminggong@wenminggon' followed by a close button 'x', a plus sign '+', and a dropdown arrow 'v'. The terminal shows the prompt 'wenminggong@wenminggong:' followed by the directory path '~/projects/ros2/project/workspace_name/src\$' in a light blue font. The rest of the terminal area is empty.

```
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*

A screenshot of a terminal window with a dark background. The window title bar shows 'wenminggong@wenminggon' and standard window controls. The terminal prompt is 'wenminggong@wenminggong:~/projects/ros2/project/workspace_name\$' with a white cursor at the end. The terminal is currently empty of any commands or output.

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

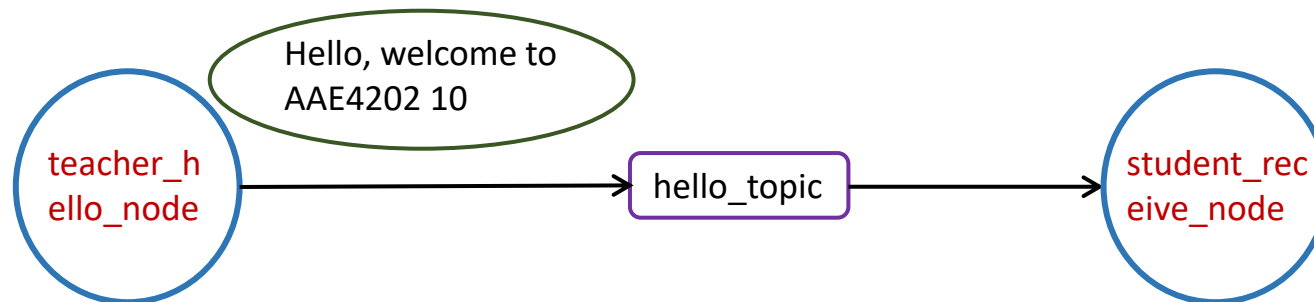
1. ROS2 Project Creation

2. Task Example

3. Project Assignment

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



Procedure

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

2. Task Example

teacher_hello_node

```
1  # import ros2 python api
2  import rclpy
3  from rclpy.node import Node
4  # import standard message format
5  from std_msgs.msg import String
6
7  # define node class
8  > class TeacherPublisher(Node): ...
41
42  def main():
43      # initialize ros2
44      rclpy.init()
45
46      # execute callbacks until shutdown
47      rclpy.spin(node=TeacherPublisher(node_name="teacher_hello_node"))
48
49      # shutdown a previously initialization
50      rclpy.shutdown()
51
52  if __name__ == '__main__':
53      main()
54
```

teacher_hello_node

```
7  # define node class
8  class TeacherPublisher(Node):
9      def __init__(self, node_name: str) -> None:
10         super().__init__(node_name)
11         # print info
12         self.get_logger().info("teacher node created!")
13
14         # create a topic publisher
15         # parameters:
16         #   msg_type: the type of message
17         #   topic: the name of topic
18         #   qos_profile: a QoSProfile for setting the communication policy or a history depth to store
19         self.hello_publisher = self.create_publisher(msg_type=String, topic="hello_topic", qos_profile=10)
20
21         # create a timer
22         # parameters:
23         #   timer_period_sec: the period of the timer
24         #   callback: a user-defined callback function that is called when the timer expires
25         self.timer = self.create_timer(timer_period_sec=1.0, callback=self._publish)
26
27         # create a counter
28         self.counter = 1
29
30     def _publish(self):
31         # create hello message
32         hello_message = String()
33         hello_message.data = "Hello, welcome to AAE4202 {}".format(self.counter)
34
35         # publish hello message
36         self.hello_publisher.publish(msg=hello_message)
37         # print info
38         self.get_logger().info("publish: {}".format(hello_message.data))
39
40         self.counter += 1
```

2. Task Example

student_receive_node

```
1  # import ros2 python api
2  ✓ import rclpy
3  from rclpy.node import Node
4
5  # import standard message format
6  from std_msgs.msg import String
7
8
9  # define node class
10 > class StudentSubscriber(Node): ...
26
27
28 ✓ def main():
29     # initialize ros2
30     rclpy.init()
31
32     # execute callbacks until shutdown
33     rclpy.spin(node=StudentSubscriber(node_name="student_receive_node"))
34
35     # shutdown a previously initialization
36     rclpy.shutdown()
37
38
39 ✓ if __name__ == '__main__':
40     main()
41
```


student_receive_node

```
8
9  # define node class
10 class StudentSubscriber(Node):
11     def __init__(self, node_name: str) -> None:
12         super().__init__(node_name)
13         # print info
14         self.get_logger().info("student node created!")
15
16         # create a topic subscriber
17         # parameters:
18         #   # msg_type: the type of message
19         #   # topic: the name of topic
20         #   # qos_profile: a QoSProfile for setting the communication policy or a history depth to store
21         self.hello_subscriber = self.create_subscription(msg_type=String, topic="hello_topic", callback=self._subscribe, qos_profile=1)
22
23     def _subscribe(self, hello_message):
24         # print info
25         self.get_logger().info("received message: {}".format(hello_message.data))
26
```

2. Task Example

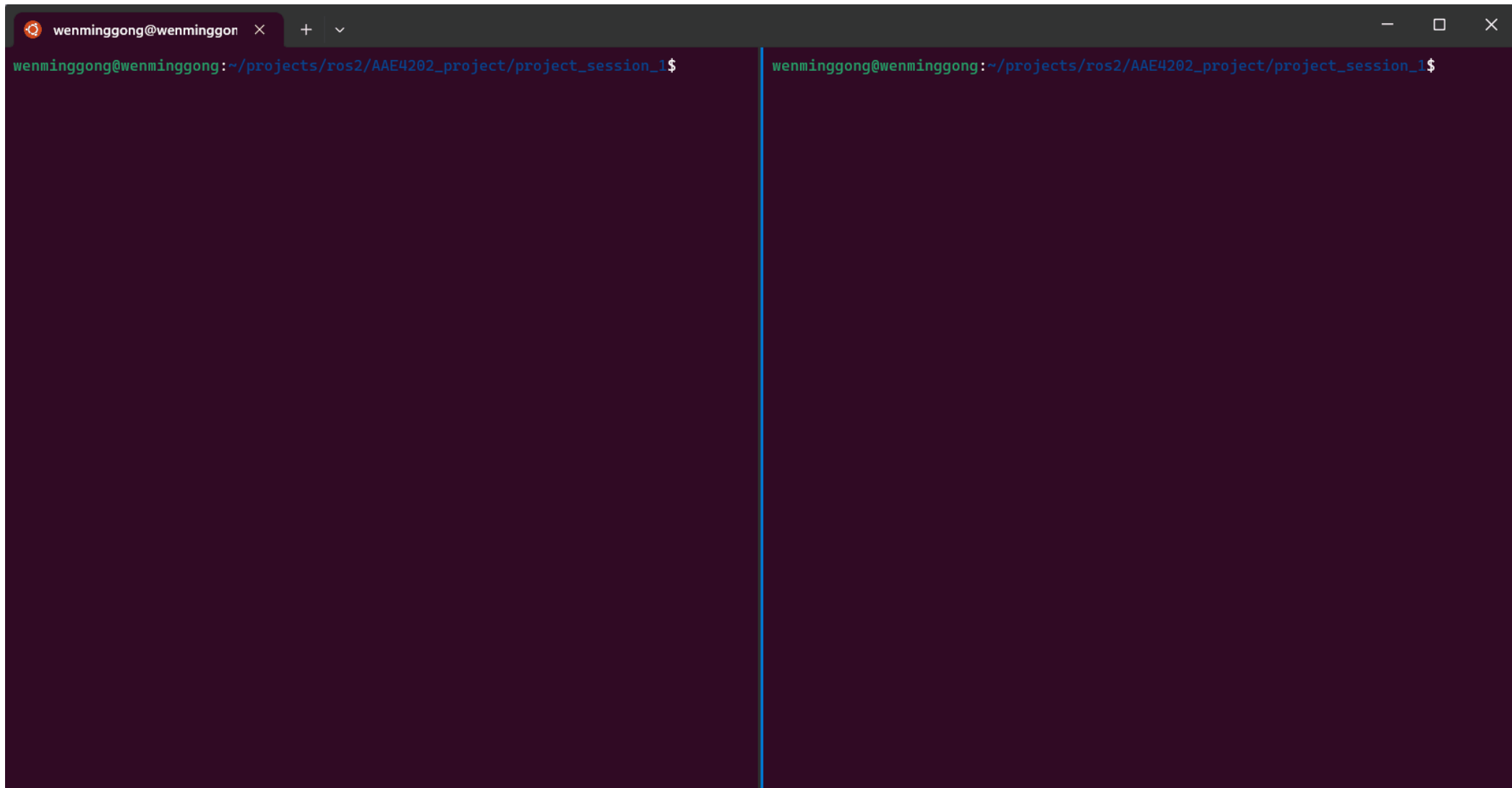
modify configuration file: setup.py

```
setup.py X
src > teacher_student_pkg > setup.py > ...
1  from setuptools import find_packages, setup
2
3  package_name = 'teacher_student_pkg'
4
5  setup(
6      name=package_name,
7      version='0.0.0',
8      packages=find_packages(exclude=['test']),
9      data_files=[
10         ('share/ament_index/resource_index/packages',
11          ['resource/' + package_name]),
12         ('share/' + package_name, ['package.xml']),
13     ],
14     install_requires=['setuptools'],
15     zip_safe=True,
16     maintainer='wenminggong',
17     maintainer_email='nandalmj@163.com',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23             'teacher_hello_node = teacher_student_pkg.teacher_hello_node:main',
24             'student_receive_node = teacher_student_pkg.student_receive_node:main'
25         ],
26     },
27 )
```

2. Task Example

Slide 19 of 28

Experimental Results



1. ROS2 Project Creation

2. Task Example

3. Project Assignment

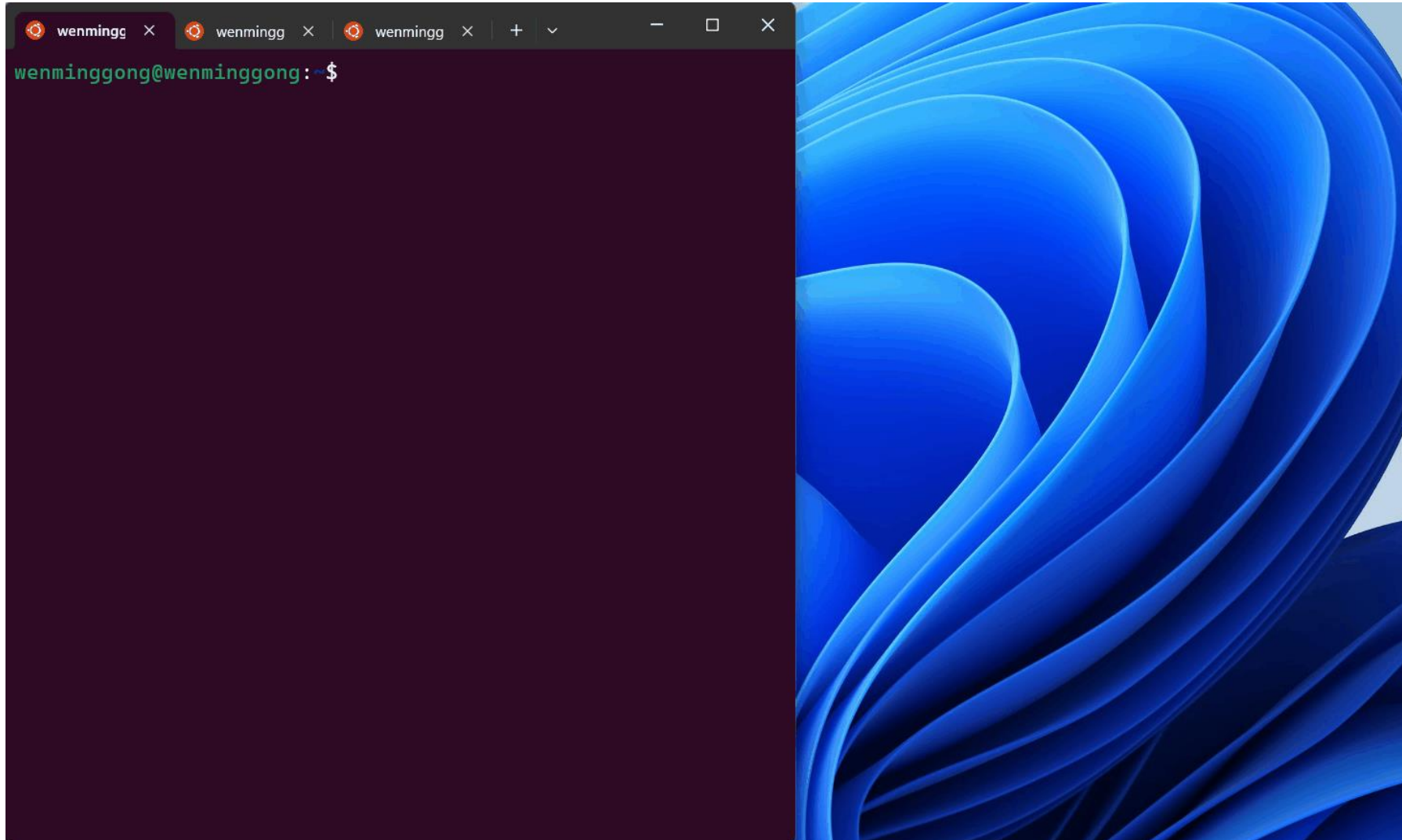
Prerequisites

- 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
 - turtlesim_teleop_key: control the turtle in the simulator by key board

3. Project Assignment

Slide 22 of 28

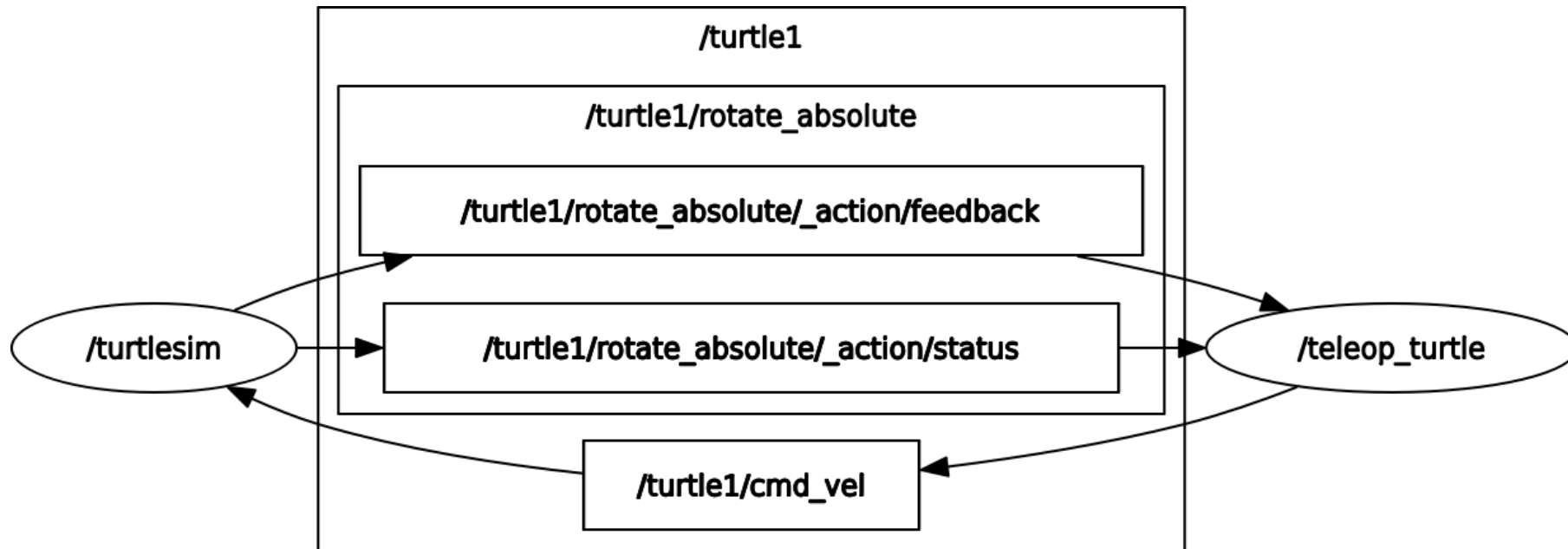
Prerequisites



Prerequisites

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

rqt_graph



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
- 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
 - Only one submission per group
 - Please submit the report in pdf format and sign the names of all members

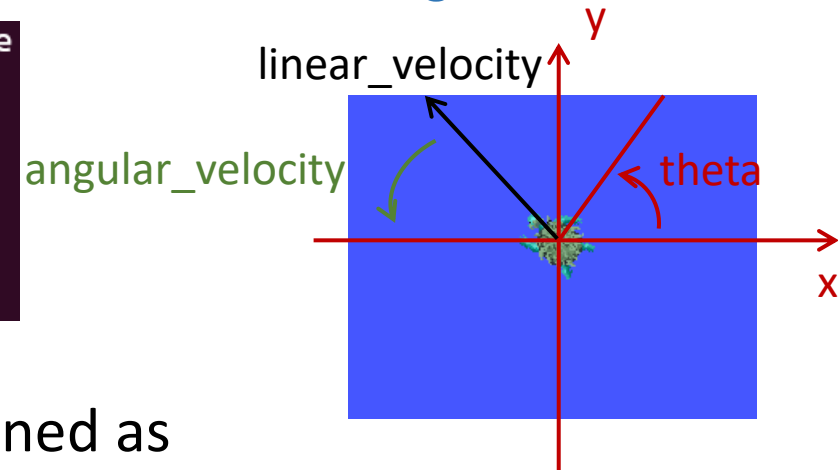
3. Project Assignment

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

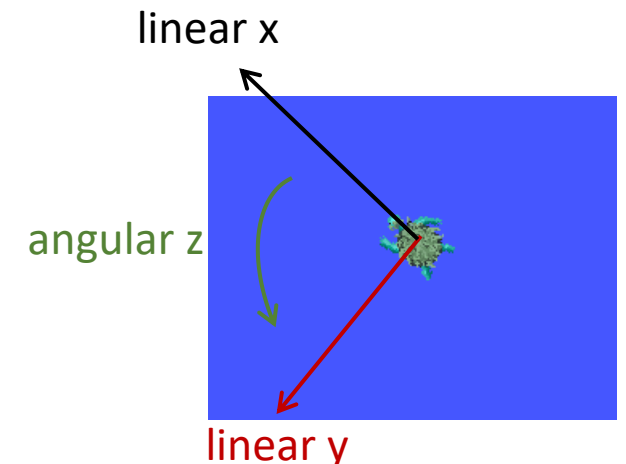
float32 linear_velocity
float32 angular_velocity
```



- 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
```



Task Description

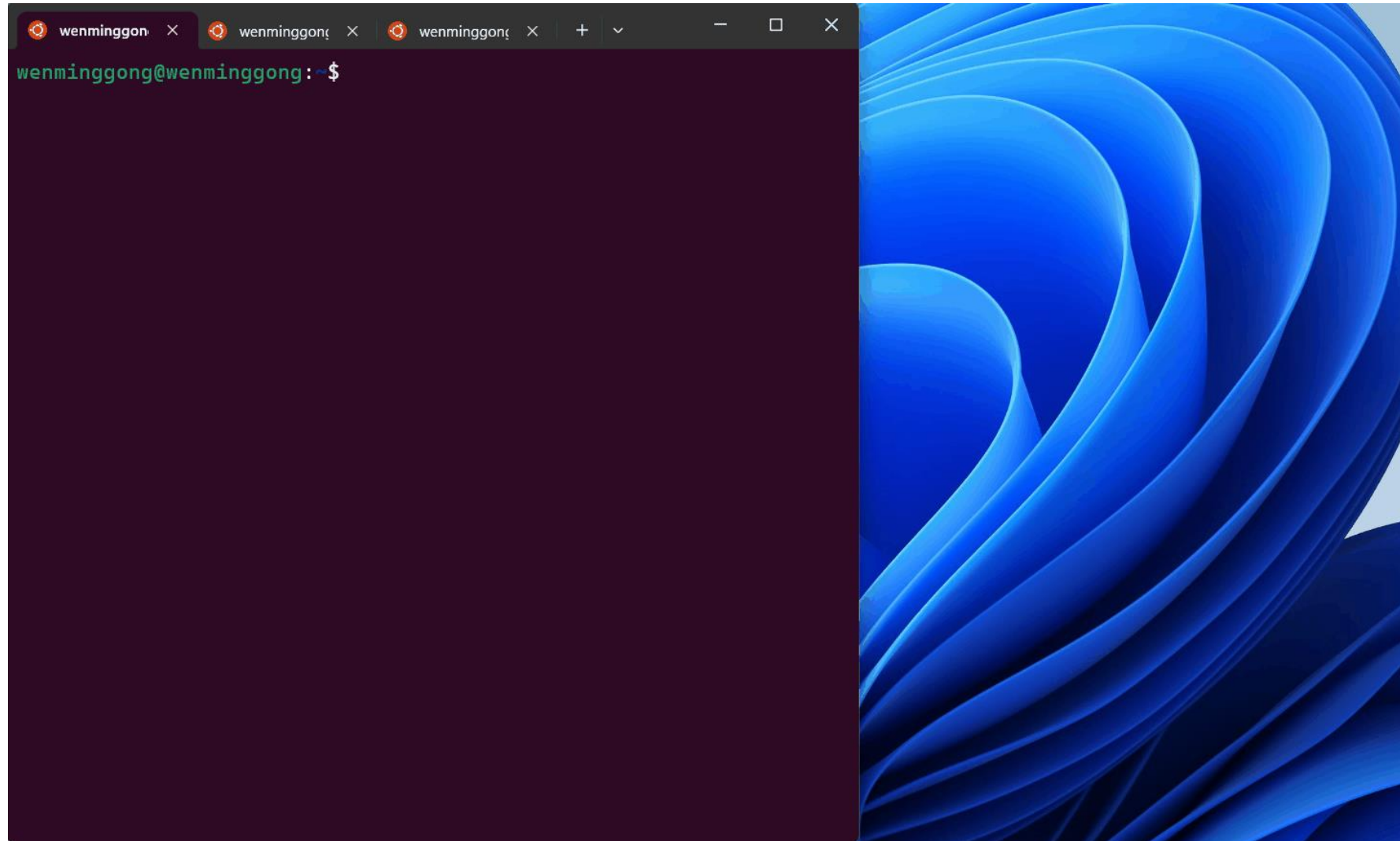
- Message dependencies need to be added when the package is created

```
# 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

Slide 27 of 28

Demonstration



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