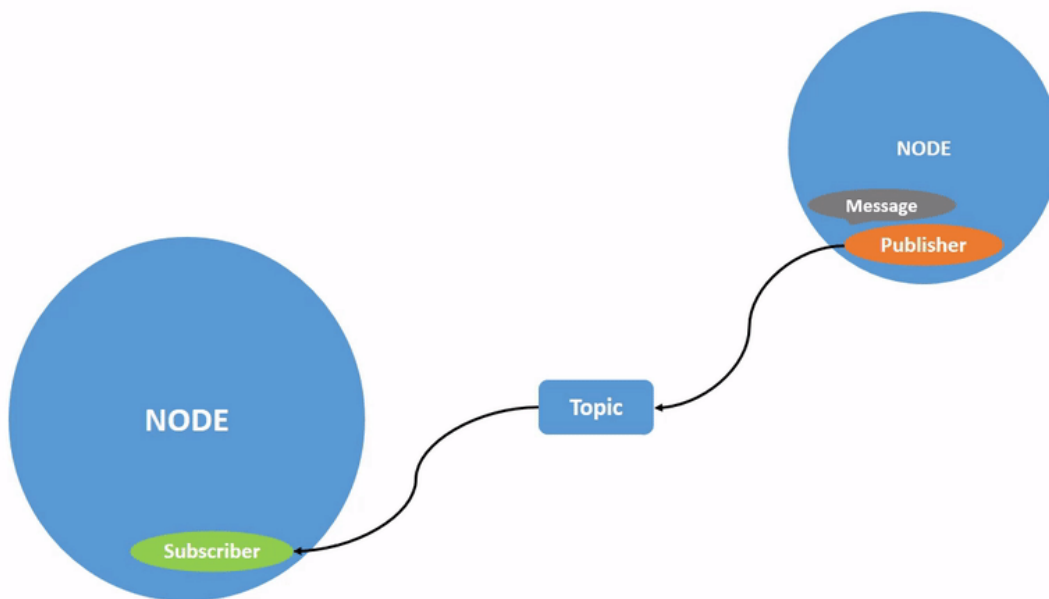


8.ROS2 topic communication subscriber

1. Introduction to topic communication

Topic communication is the most frequently used communication method in ROS2. A publisher publishes data on a specified topic, and subscribers can receive the data as long as they subscribe to the data on the topic.

Topic communication is based on the publish/subscribe model, as shown in the figure:



The characteristics of topic data transmission are from one node to another node. The object sending data is called **Publisher**, and the object receiving data is called **Subscriber**. Each topic needs to have a name, the transmitted data also needs to have a fixed data type.

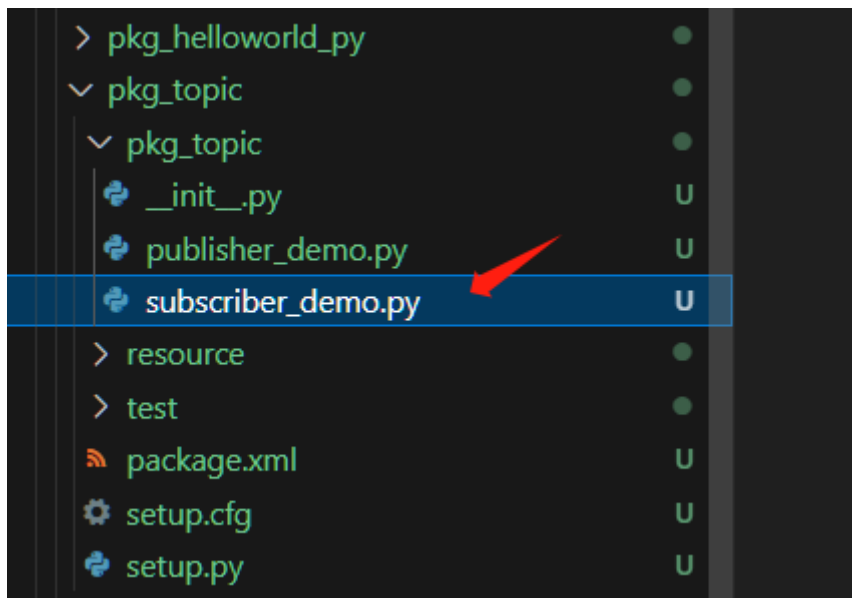
Next, we will explain how to use Python language to implement topic communication between nodes.

This case is located in the factory docker container. The source code location is:

```
/root/yahboomcar_ros2_ws/yahboomcar_ws/src/pkg_topic
```

2. Subscriber realization

Create a new file [subscriber_demo.py] in the same directory as [publisher_demo.py]

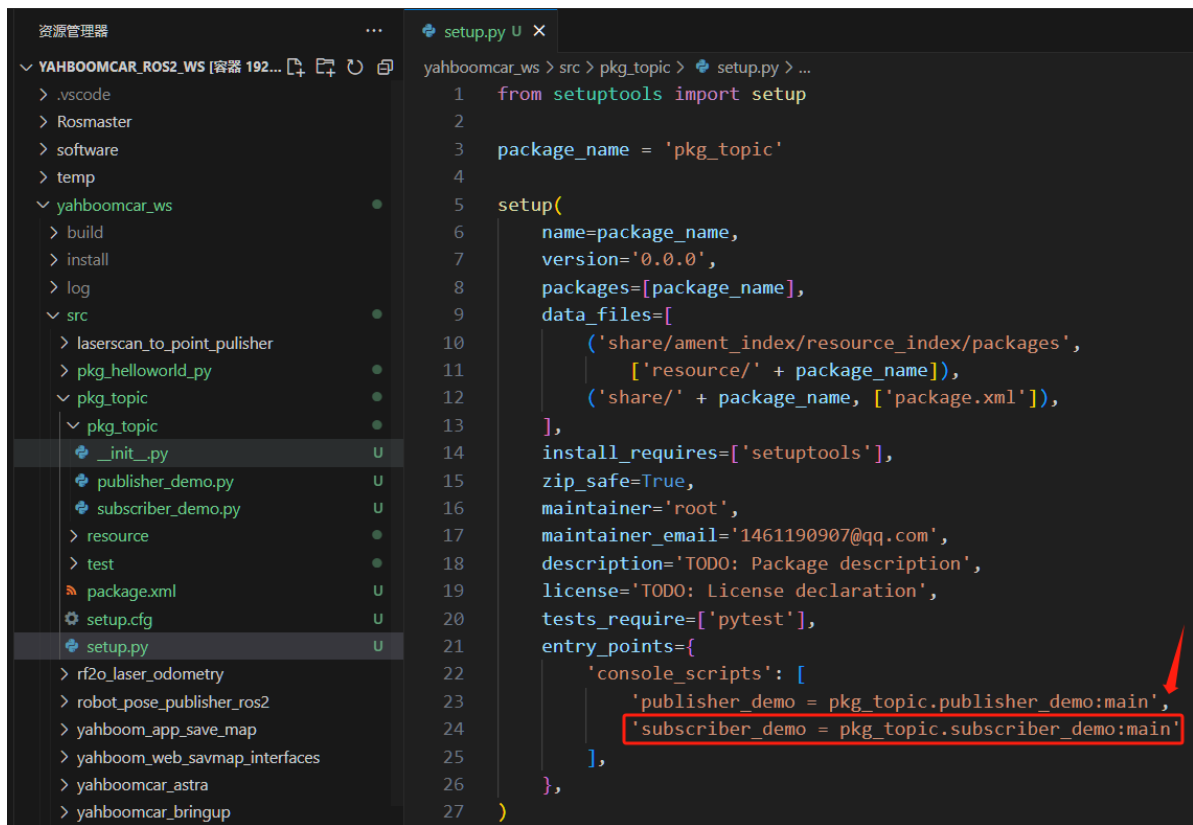


Next edit [subscriber_demo.py] to implement the subscriber function and add the following code:

```
#Import related libraries
import rclpy
from rclpy.node import Node
from std_msgs.msg import String

class Topic_Sub(Node):
    def __init__(self, name):
        super().__init__(name)
        #To create a subscriber, create_subscription is used. The parameters
        #passed in are: topic data type, topic name, callback function name, and queue
        #length.
        self.sub =
self.create_subscription(String, "/topic_demo", self.sub_callback, 1)
        #Callback function execution program: print the received information
    def sub_callback(self, msg):
        print(msg.data)
def main():
    rclpy.init() #ROS2 Python interface initialization
    sub_demo = Topic_Sub("subscriber_node") # Create the object and initialize
    it
    rclpy.spin(sub_demo)
    sub_demo.destroy_node() #Destroy node object
    rclpy.shutdown() #Close the ROS2 Python interface
```

3. Edit configuration file



```
1 from setuptools import setup
2
3 package_name = 'pkg_topic'
4
5 setup(
6     name=package_name,
7     version='0.0.0',
8     packages=[package_name],
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='root',
17     maintainer_email='1461190907@qq.com',
18     description='TODO: Package description',
19     license='TODO: License declaration',
20     tests_require=['pytest'],
21     entry_points={
22         'console_scripts': [
23             'publisher_demo = pkg_topic.publisher_demo:main',
24             'subscriber_demo = pkg_topic.subscriber_demo:main',
25         ],
26     },
27 )
```

4. Compile workspace

```
cd ~/yahboomcar_ros2_ws/yahboomcar_ws
colcon build --packages-select pkg_topic
source install/setup.bash
```

5. Run program

The sub-terminal execution is as follows:

```
#Start publisher node
ros2 run pkg_topic publisher_demo
#Start Subscriber Node
ros2 run pkg_topic subscriber_demo
```

```
root@unbutu:~/yahboomcar_ros2_ws/yahboomcar_ws# ros2 run pkg_topic publisher_demo
```

2. 192.168.2.117 (jetson)

```
root@unbutu:~/yahboomcar_ros2_ws/yahboomcar_ws# ros2 run pkg_topic subscriber_demo
Hi,I send a message.
Hi,I send a message.
Hi,I send a message.
Hi,I send a message.
Hi,I send a message.
Hi,I send a message.
Hi,I send a message.
Hi,I send a message.
```

As shown in the figure above, the terminal running the subscriber will print the /topic_demo information published by the publisher.