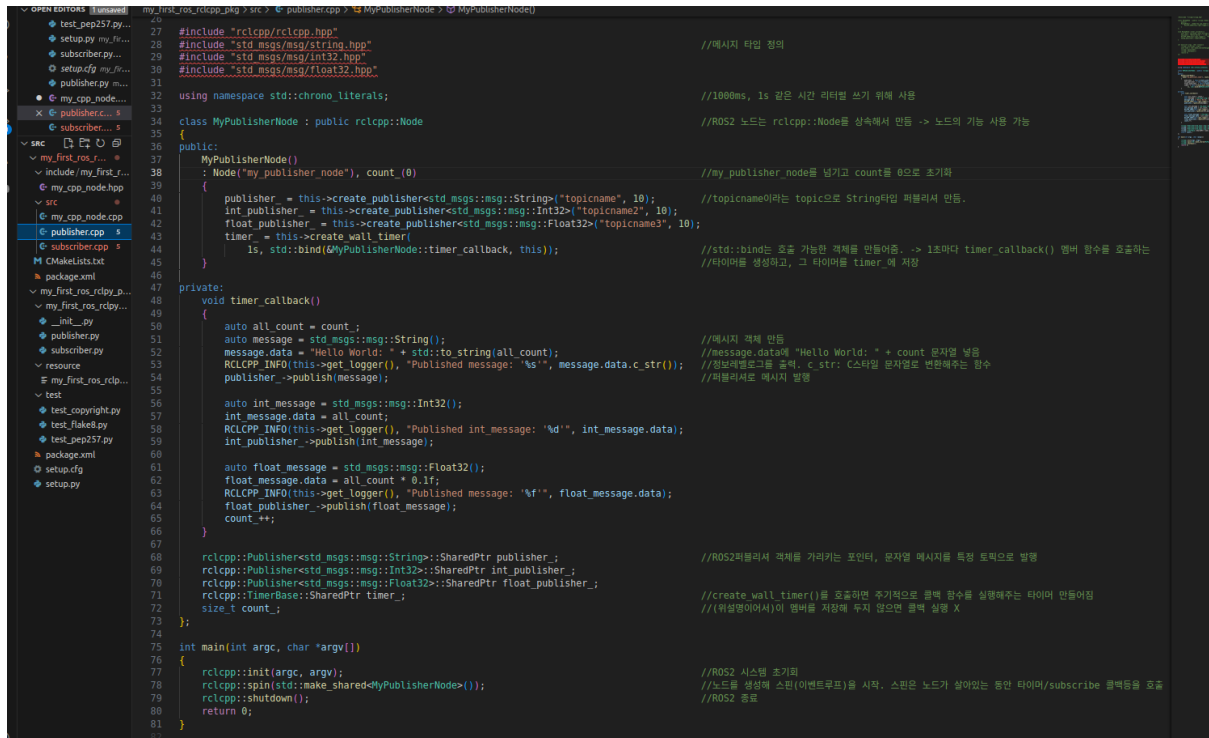


ROS 1일차 보고서

목차

1. C++ 통신
2. Python 통신
3. C++ <-> Python 통신

1. C++ 통신



```
1 27 #include "rclcpp/rclcpp.hpp"
2 28 #include "std_msgs/msg/string.hpp"
3 29 #include "std_msgs/msg/int32.hpp"
4 30 #include "std_msgs/msg/float32.hpp"
5 31
6 32 using namespace std::chrono_literals;
7 33
8 34 class MyPublisherNode : public rclcpp::Node
9 35 {
10 36 public:
11 37     MyPublisherNode()
12 38     : Node("my_publisher_node"), count_(0)
13 39     {
14 40         publisher_ = this->create_publisher<std_msgs::msg::String>("topicname", 10);
15 41         int_publisher_ = this->create_publisher<std_msgs::msg::Int32>("topicname2", 10);
16 42         float_publisher_ = this->create_publisher<std_msgs::msg::Float32>("topicname3", 10);
17 43         timer_ = this->create_wall_timer(
18 44             1s, std::bind(&MyPublisherNode::timer_callback, this));
19 45     }
20 46 private:
21 47     void timer_callback()
22 48     {
23 49         auto all_count = count_;
24 50         auto message = std_msgs::msg::String();
25 51         message.data = "Hello World: " + std::to_string(all_count);
26 52         RCLCPP_INFO(this->get_logger(), "Published message: '%s'", message.data.c_str());
27 53         publisher_->publish(message);
28 54
29 55         auto int_message = std_msgs::msg::Int32();
30 56         int_message.data = all_count;
31 57         RCLCPP_INFO(this->get_logger(), "Published int_message: '%d'", int_message.data);
32 58         int_publisher_->publish(int_message);
33 59
34 60         auto float_message = std_msgs::msg::Float32();
35 61         float_message.data = all_count * 0.1f;
36 62         RCLCPP_INFO(this->get_logger(), "Published message: '%f'", float_message.data);
37 63         float_publisher_->publish(float_message);
38 64         count_++;
39 65     }
40 66
41 67     rclcpp::Publisher<std_msgs::msg::String>::SharedPtr publisher_;
42 68     rclcpp::Publisher<std_msgs::msg::Int32>::SharedPtr int_publisher_;
43 69     rclcpp::Publisher<std_msgs::msg::Float32>::SharedPtr float_publisher_;
44 70     rclcpp::TimerBase::SharedPtr timer_;
45 71     size_t count_;
46 72 };
47 73
48 74 int main(int argc, char *argv[])
49 75 {
50 76     rclcpp::init(argc, argv);
51 77     rclcpp::spin(std::make_shared<MyPublisherNode>());
52 78     rclcpp::shutdown();
53 79     return 0;
54 80 }
55 81
```

기존 예제 코드에서 헤더파일을 int32와 float32를 호출해 int와 float 를 사용할 수 있도록 한다.

예제코드 형식에 맞춰 int와 float를 topic이름을 다르게 설정해 데이터를 전달한다.

각 int_message, float_message를 선언해 퍼블리셔로 메시지를 발행한다.

```

1  /*#include "rclcpp/rclcpp.hpp"
21
22 #include "rclcpp/rclcpp.hpp"
23 #include "std_msgs/msg/string.hpp"
24 #include "std_msgs/msg/int32.hpp"
25 #include "std_msgs/msg/float32.hpp"
26
27 class MySubscriberNode : public rclcpp::Node
28 {
29 public:
30     MySubscriberNode()
31     : Node("my_subscriber_node")
32     {
33         // subscriber 생성
34         subscription = this->create_subscription<std_msgs::msg::String>(
35             "topicname", 10,
36             std::bind(&MySubscriberNode::topic_callback, this, std::placeholders::_1));
37
38         int_subscription = this->create_subscription<std_msgs::msg::Int32>(
39             "topicname2", 10,
40             std::bind(&MySubscriberNode::int_callback, this, std::placeholders::_1));
41
42         float_subscription = this->create_subscription<std_msgs::msg::Float32>(
43             "topicname3", 10,
44             std::bind(&MySubscriberNode::float_callback, this, std::placeholders::_1));
45
46     };
47
48 private:
49     void topic_callback(const std_msgs::msg::String::SharedPtr msg)
50     {
51         RCLCPP_INFO(this->get_logger(), "Received message: '%s'", msg->data.c_str());
52     }
53
54     void int_callback(const std_msgs::msg::Int32::SharedPtr msg){
55         RCLCPP_INFO(this->get_logger(), "Received message: '%d'", msg->data);
56     }
57
58     void float_callback(const std_msgs::msg::Float32::SharedPtr msg){
59         RCLCPP_INFO(this->get_logger(), "Received message: '%f'", msg->data);
60     }
61
62     rclcpp::Subscription<std_msgs::msg::String::SharedPtr> subscription_;
63     rclcpp::Subscription<std_msgs::msg::Int32::SharedPtr> int_subscription_;
64     rclcpp::Subscription<std_msgs::msg::Float32::SharedPtr> float_subscription_;
65
66 };
67
68 int main(int argc, char *argv[])
69 {
70     rclcpp::init(argc, argv);
71     rclcpp::spin(std::make_shared<MySubscriberNode>());
72     rclcpp::shutdown();
73     return 0;
74 }
75

```

Publisher에서 설정한 topic이름을 subscriber에서도 그대로 써준다.

Topic_callback도 예제형식에 맞춰 int, float의 callback 함수도 만들어 준다. 받은 메시지의 로그를 출력할 수 있게 해준다.

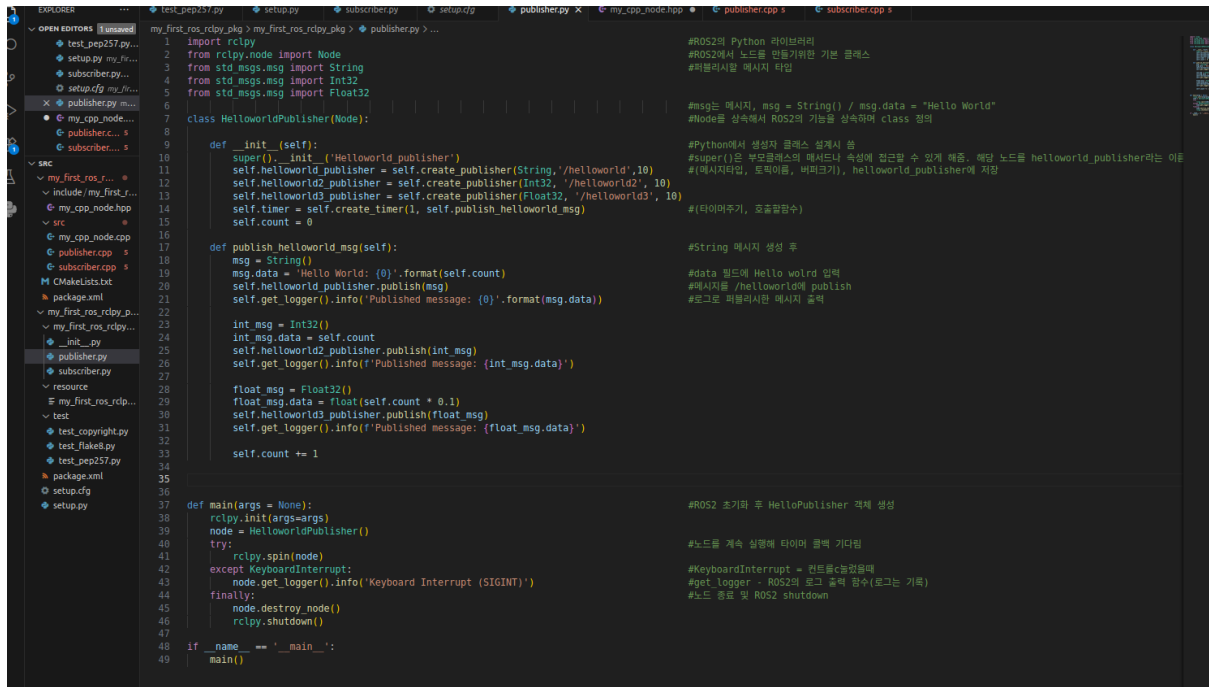
```

kmj@kmj:~/Desktop/hw2 93x27
[INFO] [1757912713.419196795] [my_publisher_node]: Published message: '2.400000'
[INFO] [1757912714.418835167] [my_publisher_node]: Published message: 'Hello World: 25'
[INFO] [1757912714.419026476] [my_publisher_node]: Published int_message: '25'
[INFO] [1757912714.419054308] [my_publisher_node]: Published message: '2.500000'
[INFO] [1757912715.418800415] [my_publisher_node]: Published message: 'Hello World: 26'
[INFO] [1757912715.418967589] [my_publisher_node]: Published int_message: '26'
[INFO] [1757912715.418999639] [my_publisher_node]: Published message: '2.600000'
[INFO] [1757912716.418817296] [my_publisher_node]: Published message: 'Hello World: 27'
[INFO] [1757912716.418982325] [my_publisher_node]: Published int_message: '27'
[INFO] [1757912716.419002172] [my_publisher_node]: Published message: '2.700000'
[INFO] [1757912717.418801277] [my_publisher_node]: Published message: 'Hello World: 28'
[INFO] [1757912717.419018158] [my_publisher_node]: Published int_message: '28'
[INFO] [1757912717.419048260] [my_publisher_node]: Published message: '2.800000'
[INFO] [1757912718.418836000] [my_publisher_node]: Published message: 'Hello World: 29'
[INFO] [1757912718.419006028] [my_publisher_node]: Published int_message: '29'
[INFO] [1757912718.419021638] [my_publisher_node]: Published message: '2.900000'
[INFO] [1757912719.419087446] [my_publisher_node]: Published message: 'Hello World: 30'
[INFO] [1757912719.419188611] [my_publisher_node]: Published int_message: '30'
[INFO] [1757912719.419232964] [my_publisher_node]: Published message: '3.000000'
[INFO] [1757912720.418891919] [my_publisher_node]: Published message: 'Hello World: 31'
[INFO] [1757912720.418989616] [my_publisher_node]: Published int_message: '31'
[INFO] [1757912720.419017848] [my_publisher_node]: Published message: '3.100000'
[INFO] [1757912721.419033503] [my_publisher_node]: Published message: 'Hello World: 32'
[INFO] [1757912721.419312048] [my_publisher_node]: Published int_message: '32'
[INFO] [1757912721.419345892] [my_publisher_node]: Published message: '3.200000'
^C[INFO] [1757912721.944942778] [rclcpp]: signal_handler(signum=2)
kmj@kmj:~/Desktop/hw2$

kmj@kmj:~/Desktop/hw2 94x27
[INFO] [1757912713.419631898] [my_subscriber_node]: Received message: '2.400000'
[INFO] [1757912714.419252790] [my_subscriber_node]: Received message: 'Hello World: 25'
[INFO] [1757912714.419375269] [my_subscriber_node]: Received message: '25'
[INFO] [1757912714.419413140] [my_subscriber_node]: Received message: '2.500000'
[INFO] [1757912715.419182141] [my_subscriber_node]: Received message: 'Hello World: 26'
[INFO] [1757912715.419283791] [my_subscriber_node]: Received message: '26'
[INFO] [1757912715.419324738] [my_subscriber_node]: Received message: '2.600000'
[INFO] [1757912716.419239637] [my_subscriber_node]: Received message: 'Hello World: 27'
[INFO] [1757912716.419335307] [my_subscriber_node]: Received message: '27'
[INFO] [1757912716.419373859] [my_subscriber_node]: Received message: '2.700000'
[INFO] [1757912717.419221514] [my_subscriber_node]: Received message: 'Hello World: 28'
[INFO] [1757912717.419315561] [my_subscriber_node]: Received message: '28'
[INFO] [1757912717.419350767] [my_subscriber_node]: Received message: '2.800000'
[INFO] [1757912718.419246659] [my_subscriber_node]: Received message: 'Hello World: 29'
[INFO] [1757912718.419356435] [my_subscriber_node]: Received message: '29'
[INFO] [1757912718.419396971] [my_subscriber_node]: Received message: '2.900000'
[INFO] [1757912719.419670283] [my_subscriber_node]: Received message: 'Hello World: 30'
[INFO] [1757912719.419746105] [my_subscriber_node]: Received message: '30'
[INFO] [1757912719.419772274] [my_subscriber_node]: Received message: '3.000000'
[INFO] [1757912720.419215338] [my_subscriber_node]: Received message: 'Hello World: 31'
[INFO] [1757912720.419309835] [my_subscriber_node]: Received message: '31'
[INFO] [1757912720.419351884] [my_subscriber_node]: Received message: '3.100000'
[INFO] [1757912721.419685318] [my_subscriber_node]: Received message: 'Hello World: 32'
[INFO] [1757912721.419870485] [my_subscriber_node]: Received message: '32'
[INFO] [1757912721.419929976] [my_subscriber_node]: Received message: '3.200000'
^C[INFO] [1757912726.435165188] [rclcpp]: signal_handler(signum=2)
kmj@kmj:~/Desktop/hw2$

```

2. Python 통신



```
1 import rclpy
2 from rclpy.node import Node
3 from std_msgs.msg import String
4 from std_msgs.msg import Int32
5 from std_msgs.msg import Float32
6
7 class HelloWorldPublisher(Node):
8
9
10     def __init__(self):
11         super().__init__('HelloWorldPublisher')
12         self.helloworld_publisher = self.create_publisher(String, '/helloworld', 10)
13         self.helloworld2_publisher = self.create_publisher(Int32, '/helloworld2', 10)
14         self.helloworld3_publisher = self.create_publisher(Float32, '/helloworld3', 10)
15         self.timer = self.create_timer(1, self.publish_helloworld_msg)
16         self.count = 0
17
18     def publish_helloworld_msg(self):
19         msg = String()
20         msg.data = 'Hello World: {}'.format(self.count)
21         self.helloworld_publisher.publish(msg)
22         self.get_logger().info('Published message: {}'.format(msg.data))
23
24         int_msg = Int32()
25         int_msg.data = self.count
26         self.helloworld2_publisher.publish(int_msg)
27         self.get_logger().info('Published message: {}'.format(int_msg.data))
28
29         float_msg = Float32()
30         float_msg.data = float(self.count * 0.1)
31         self.helloworld3_publisher.publish(float_msg)
32         self.get_logger().info('Published message: {}'.format(float_msg.data))
33
34         self.count += 1
35
36
37 def main(args = None):
38     rclpy.init(args=args)
39     node = HelloWorldPublisher()
40     try:
41         rclpy.spin(node)
42     except KeyboardInterrupt:
43         node.get_logger().info('Keyboard Interrupt (SIGINT)')
44     finally:
45         node.destroy_node()
46         rclpy.shutdown()
47
48 if __name__ == '__main__':
49     main()
```

#ROS2의 Python 라이브러리
#ROS2에서 노드를 만들기위한 기본 클래스
#퍼블리셔를 메시지 타입

#msg는 메시지, msg = String() / msg.data = "Hello World"
#Node를 상속해서 ROS2의 기능을 상속하여 class 정의

#Python에서 생성자 클래스 설정시 self
#super()은 부모클래스의 메서드나 속성에 접근할 수 있게 해줌, 해당 노드를 helloworld_publisher라는 이름
#(메시지타입, 토픽이름, 버퍼크기), helloworld_publisher에 저장

#(타이머주기, 호출함수)

#String 메시지 생성 후

#data 필드에 Hello world 입력
#메시지를 /helloworld에 publish
#로그로 퍼블리셔한 메시지 출력

#ROS2 초기화 후 HelloWorldPublisher 객체 생성

#노드를 계속 실행해 타이머 콜백 기다림

#KeyboardInterrupt = 컨트롤+c눌렀을때
#get_logger() - ROS2의 로그 출력 함수(로그는 기록)
#노드 종료 및 ROS2 shutdown

예제코드에서 int와 float의 헤더파일을 추가로 선언해준다.

Topic 이름을 int와 float 모두 다르게 설정을 해주고 버퍼크기는 10으로 통일한다.

Int형 메시지를 전달하는 int_msg, float형 메시지를 전달하는 float_msg를 만들어준다. float형은 0.1을 곱해주어 실수를 출력할 수 있게 해준다.

```
EXPLORER ... test_pep257.py setup.py subscriber.py x setup.cfg publisher.py my_cpp_node.hpp publisher.cpp s subscriber.cpp s
my_first_ros_rcloy pkg > my_first_ros_rcloy_pkg > subscriber.py > HelloworldSubscriber > _init_
1 import rclpy
2 from rclpy.node import Node
3 from std_msgs.msg import String
4 from std_msgs.msg import Int32
5 from std_msgs.msg import Float32
6
7 class HelloworldSubscriber(Node):
8     #ROS2 노드 정의하는 클래스
9
10     def __init__(self):
11         super().__init__('Helloworld subscriber')
12         self.subscription = self.create_subscription(
13             String,
14             '/helloworld',
15             self.subscription_callback,
16             10
17         )
18
19     def subscription_callback(self, msg):
20         self.get_logger().info(f'Received message: {msg.data}')
21
22     self.int_subscription = self.create_subscription(
23         Int32,
24         '/helloworld2',
25         self.int_subscription_callback,
26         10
27     )
28
29     self.float_subscription = self.create_subscription(
30         Float32,
31         '/helloworld3',
32         self.float_subscription_callback,
33         10
34     )
35
36     def subscription_callback(self, msg):
37         self.get_logger().info(f'Received message: {msg.data}')
38
39     def int_subscription_callback(self, int_msg):
40         self.get_logger().info(f'Received message: {int_msg.data}')
41
42     def float_subscription_callback(self, float_msg):
43         self.get_logger().info(f'Received message: {float_msg.data}')
44
45 def main(args=None):
46     rclpy.init(args=args)
47     node = HelloworldSubscriber()
48     try:
49         node.spin()
50     except KeyboardInterrupt:
51         node.get_logger().info('Keyboard Interrupt (SIGINT)')
52     finally:
53         node.destroy_node()
54         rclpy.shutdown()
55
56 if __name__ == '__main__':
57     main()
```

마찬가지로 예제코드에서 int와 float 헤더파일을 선언한다.

메시지타입, 토픽이름, 콜백함수, 버퍼크기에 맞게 객체를 지정해준다.

콜백함수를 int와 float 데이터에 맞게 출력할 수 있도록해준다.

```
lmi@lmi:/Desktop/hw2 $ ros2 run my_cpp_node helloworld_subscriber
[INFO] [1757915471.762497071] [helloworld_publisher]: Published message: 126
[INFO] [1757915471.762755546] [helloworld_publisher]: Published message: 12.600000000000001
[INFO] [1757915472.762311941] [helloworld_publisher]: Published message: Hello World: 127
[INFO] [1757915472.762624011] [helloworld_publisher]: Published message: 127
[INFO] [1757915472.762873677] [helloworld_publisher]: Published message: 12.700000000000001
[INFO] [1757915473.761684783] [helloworld_publisher]: Published message: Hello World: 128
[INFO] [1757915473.762083036] [helloworld_publisher]: Published message: 128
[INFO] [1757915473.762274826] [helloworld_publisher]: Published message: 12.8
[INFO] [1757915474.762439262] [helloworld_publisher]: Published message: Hello World: 129
[INFO] [1757915474.762788455] [helloworld_publisher]: Published message: 129
[INFO] [1757915474.763329804] [helloworld_publisher]: Published message: 12.9
[INFO] [1757915475.763452464] [helloworld_publisher]: Published message: Hello World: 130
[INFO] [1757915475.763314322] [helloworld_publisher]: Published message: 130
[INFO] [1757915475.762548581] [helloworld_publisher]: Published message: 13.0
[INFO] [1757915476.76222710] [helloworld_publisher]: Published message: Hello World: 131
[INFO] [1757915476.762518787] [helloworld_publisher]: Published message: 131
[INFO] [1757915476.762768682] [helloworld_publisher]: Published message: 13.100000000000001
[INFO] [1757915477.762256959] [helloworld_publisher]: Published message: Hello World: 132
[INFO] [1757915477.762689212] [helloworld_publisher]: Published message: 132
[INFO] [1757915477.76319535] [helloworld_publisher]: Published message: 13.200000000000001
[INFO] [1757915478.762893216] [helloworld_publisher]: Published message: Hello World: 133
[INFO] [1757915478.762570550] [helloworld_publisher]: Published message: 133
[INFO] [1757915478.762833716] [helloworld_publisher]: Published message: 13.3
[INFO] [1757915479.762859717] [helloworld_publisher]: Published message: Hello World: 134
[INFO] [1757915479.762413976] [helloworld_publisher]: Published message: 134
[INFO] [1757915479.762638015] [helloworld_publisher]: Published message: 13.4

lmi@lmi:/Desktop/hw2 $ ros2 run my_cpp_node helloworld_subscriber
[INFO] [1757915471.763081581] [helloworld_subscriber]: Received message: 126
[INFO] [1757915471.763376816] [helloworld_subscriber]: Received message: 12.600000381469727
[INFO] [1757915472.763105361] [helloworld_subscriber]: Received message: Hello World: 127
[INFO] [1757915472.763596262] [helloworld_subscriber]: Received message: 127
[INFO] [1757915472.763863704] [helloworld_subscriber]: Received message: 12.69999989265137
[INFO] [1757915473.761940346] [helloworld_subscriber]: Received message: Hello World: 128
[INFO] [1757915473.762759915] [helloworld_subscriber]: Received message: 128
[INFO] [1757915473.763182702] [helloworld_subscriber]: Received message: 12.800000190734863
[INFO] [1757915474.762991297] [helloworld_subscriber]: Received message: Hello World: 129
[INFO] [1757915474.763435448] [helloworld_subscriber]: Received message: 129
[INFO] [1757915474.763889214] [helloworld_subscriber]: Received message: 12.899999618530273
[INFO] [1757915475.762167631] [helloworld_subscriber]: Received message: Hello World: 130
[INFO] [1757915475.762683686] [helloworld_subscriber]: Received message: 130
[INFO] [1757915475.762932656] [helloworld_subscriber]: Received message: 13.0
[INFO] [1757915476.762486726] [helloworld_subscriber]: Received message: Hello World: 131
[INFO] [1757915476.763188855] [helloworld_subscriber]: Received message: 131
[INFO] [1757915476.763438128] [helloworld_subscriber]: Received message: 13.100000381469727
[INFO] [1757915477.762435461] [helloworld_subscriber]: Received message: Hello World: 132
[INFO] [1757915477.762995013] [helloworld_subscriber]: Received message: 132
[INFO] [1757915477.76346747] [helloworld_subscriber]: Received message: 13.19999989265137
[INFO] [1757915478.762637449] [helloworld_subscriber]: Received message: Hello World: 133
[INFO] [1757915478.763648204] [helloworld_subscriber]: Received message: 133
[INFO] [1757915478.763989414] [helloworld_subscriber]: Received message: 13.300000190734863
[INFO] [1757915479.762276331] [helloworld_subscriber]: Received message: Hello World: 134
[INFO] [1757915479.762676348] [helloworld_subscriber]: Received message: 134
[INFO] [1757915479.763013835] [helloworld_subscriber]: Received message: 13.399999618530273
```

3. C++ <-> Python 통신

The image shows two code editors side-by-side, displaying ROS2 code for a publisher and subscriber. The left editor shows the C++ implementation, and the right editor shows the Python implementation.

C++ Implementation (Left):

```

1 #include <rcpputils/rcutils.hpp>
2 #include <string>
3 #include <memory>
4 #include <chrono>
5 #include <thread>
6 #include <atomic>
7 #include <functional>
8 #include <future>
9 #include <stdexcept>
10 #include <string_view>
11 #include <vector>
12 #include <map>
13 #include <unordered_map>
14 #include <unordered_set>
15 #include <algorithm>
16 #include <iterator>
17 #include <numeric>
18 #include <tuple>
19 #include <optional>
20 #include <variant>
21 #include <any>
22 #include <string_view>
23 #include <string_view>
24 #include <string_view>
25 #include <string_view>
26 #include <string_view>
27 #include <string_view>
28 #include <string_view>
29 #include <string_view>
30 #include <string_view>
31 #include <string_view>
32 #include <string_view>
33 #include <string_view>
34 #include <string_view>
35 #include <string_view>
36 #include <string_view>
37 #include <string_view>
38 #include <string_view>
39 #include <string_view>
40 #include <string_view>
41 #include <string_view>
42 #include <string_view>
43 #include <string_view>
44 #include <string_view>
45 #include <string_view>
46 #include <string_view>
47 #include <string_view>
48 #include <string_view>
49 #include <string_view>
50 #include <string_view>
51 #include <string_view>
52 #include <string_view>
53 #include <string_view>
54 #include <string_view>
55 #include <string_view>
56 #include <string_view>
57 #include <string_view>
58 #include <string_view>
59 #include <string_view>
60 #include <string_view>
61 #include <string_view>
62 #include <string_view>
63 #include <string_view>
64 #include <string_view>
65 #include <string_view>
66 #include <string_view>
67 #include <string_view>
68 #include <string_view>
69 #include <string_view>
70 #include <string_view>
71 #include <string_view>
72 #include <string_view>
73 #include <string_view>
74 #include <string_view>
75 #include <string_view>
76 #include <string_view>
77 #include <string_view>
78 #include <string_view>
79 #include <string_view>
80 #include <string_view>
81 #include <string_view>
82 #include <string_view>
83 #include <string_view>
84 #include <string_view>
85 #include <string_view>
86 #include <string_view>
87 #include <string_view>
88 #include <string_view>
89 #include <string_view>
90 #include <string_view>
91 #include <string_view>
92 #include <string_view>
93 #include <string_view>
94 #include <string_view>
95 #include <string_view>
96 #include <string_view>
97 #include <string_view>
98 #include <string_view>
99 #include <string_view>
100 #include <string_view>

```

Python Implementation (Right):

```

1 import rclpy
2 from rclpy.node import Node
3 from std_msgs.msg import String
4 from std_msgs.msg import Int32
5 from std_msgs.msg import Float32
6
7 class HelloWorldSubscriber(Node):
8     """ROS2 노드 정의하는 클래스"""
9
10     def __init__(self):
11         super().__init__('HelloWorldSubscriber')
12         self.subscription = self.create_subscription(
13             String,
14             'topicname',
15             self.callback,
16             10)
17
18     def callback(self, msg):
19         self.int_subscription = self.create_subscription(
20             Int32,
21             'topicname2',
22             self.int_callback,
23             10)
24
25     def int_callback(self, msg):
26         self.float_subscription = self.create_subscription(
27             Float32,
28             'topicname3',
29             self.float_callback,
30             10)
31
32     def subscribe_topic_message(self, msg):
33         self.get_logger().info('Received message: {}'.format(msg.data))
34
35     def int_topic_message(self, int_msg):
36         self.get_logger().info('Received message: (int_msg.data)')
37
38     def float_topic_message(self, float_msg):
39         self.get_logger().info('Received message: (float_msg.data)')
40
41 def main(args=None):
42     rclpy.init(args=args)
43     node = HelloWorldSubscriber()
44     try:
45         rclpy.spin(node)
46     except KeyboardInterrupt:
47         node.get_logger().info('Keyboard Interrupt (SIGINT)')
48     finally:
49         node.destroy_node()
50         rclpy.shutdown()
51
52 if __name__ == '__main__':
53     main()

```

원래 코드에서 토픽이름만 맞춰주면 서로 publish와 subscribe가 가능하다.

The image shows two terminal windows side-by-side, displaying the output of the ROS2 publisher and subscriber nodes.

Terminal 1 (Left):

```

Finished <<< my_first_ros_rclpy_pkg [0.55s]
Summary: 2 packages finished [0.71s]
[INFO] [1757915624.081948889] [my_publisher_node]: Published message: 'Hello World: 0'
[INFO] [1757915624.082064891] [my_publisher_node]: Published int_message: '0'
[INFO] [1757915624.082103561] [my_publisher_node]: Published message: '0.000000'
[INFO] [1757915625.081980336] [my_publisher_node]: Published message: 'Hello World: 1'
[INFO] [1757915625.082084305] [my_publisher_node]: Published int_message: '1'
[INFO] [1757915625.082162280] [my_publisher_node]: Published message: '0.100000'
[INFO] [1757915626.081908429] [my_publisher_node]: Published message: 'Hello World: 2'
[INFO] [1757915626.082105525] [my_publisher_node]: Published int_message: '2'
[INFO] [1757915626.082140201] [my_publisher_node]: Published message: '0.200000'

```

Terminal 2 (Right):

```

Finished <<< my_first_ros_rclpy_pkg [0.51s]
Summary: 2 packages finished [0.66s]
[INFO] [1757915624.091211382] [HelloWorldSubscriber]: Received message: Hello World: 0
[INFO] [1757915624.09135650] [HelloWorldSubscriber]: Received message: 0
[INFO] [1757915624.091799593] [HelloWorldSubscriber]: Received message: 0.0
[INFO] [1757915625.083148985] [HelloWorldSubscriber]: Received message: Hello World: 1
[INFO] [1757915625.083480748] [HelloWorldSubscriber]: Received message: 1
[INFO] [1757915625.083738519] [HelloWorldSubscriber]: Received message: 0.10000000149011612
[INFO] [1757915626.08313342] [HelloWorldSubscriber]: Received message: Hello World: 2
[INFO] [1757915626.083486367] [HelloWorldSubscriber]: Received message: 2
[INFO] [1757915626.083750250] [HelloWorldSubscriber]: Received message: 0.20000000298023224

```