ROS2 day2 hw4 결과보고서

2025407012/로봇학부/송연우

목차

1. Hpp
(1) Mainwindow
(2) qnode
2. Cpp
(1) mainwindow
(2) qnode
3. 실행 화면

1. Hpp

(1) Mainwindow

헤더파일에서의 선언부입니다. 시계 방향 회전, 반시계방향 회전, 로봇팔을 그리는 함수, 회전에 쓸 회전각 등을 선언했습니다.

```
37    public:
38      Ui::MainWindowDesign* ui;
39      MainWindow(QWidget* parent = nullptr);
40      ~MainWindow();
41      QNode* qnode;
42      void label_2(QString *msg);
43      void paintEvent(QPaintEvent *e);
44      void onBtnPress();
45      void rotateJoint_CW();
46      void rotateJoint_CCW();
47      void rotate1();
48      void rotate2();
49      void rotate3();
50
51    private slots:
52      void on_pushButton_clicked();
53
54      void on_pushButton_2_clicked();
55
56
57    private:
58      void closeEvent(QCloseEvent* event);
59      int joint1_angle = 0;
60      int _degree =0;
61      int joint_degree=0;//회전각
62      int rotDirec[3]={0};
63      QTimer *Timer1;
64      QTimer *Timer2;
65    };
```

(2) Qnode

```
40  class QNode : public QThread
41  {
42    Q_OBJECT
43  public:
44    QNode();
45    ~QNode();
46    void topic_callback(const std_msgs::msg::String::SharedPtr msg);
47    void publishing(QString text);
48    //void setMessage(Qstring str);
49
50  protected:
51    void run();
52
53  private:
54    std::shared_ptr<rclcpp::Node> node;
55    rclcpp::Publisher<std_msgs::msg::String>::SharedPtr publisher;
56    rclcpp::Subscription<std_msgs::msg::String>::SharedPtr subscriber;
57
58  Q_SIGNALS:
59    void rosShutDown();
60    void receivedMessage_CW(QString msg);
61    void receivedMessage_CCW(QString msg);
62  };
```

Qnode 헤더파일 선언부분입니다. 발행하는 함수인 publishing(), 수신하는 함수인 topic_callback(), 시그널 함수 2개를 추가했습니다.

2. cpp파일

(1) mainwindow

```cpp
MainWindow::MainWindow(QWidget* parent) : QMainWindow(parent), ui(new Ui::MainWindowDesign)
{
  ui->setupUi(this);

  QIcon icon(":/ros-icon.png");
  this->setWindowIcon(icon);

  qnode = new QNode();

  QObject::connect(qnode, SIGNAL(rosShutDown()), this, SLOT(close()));
  // connect(qnode, &QNode::receivedMessage, this, [this](QString msg){
  //   ui->label_2->setText(msg);
  // });
  qnode->start();
  Timer1 = new QTimer(this);
  Timer2 = new QTimer(this);
  Timer1->setInterval(100);
  Timer2->setInterval(100);
  connect(qnode, &QNode::receivedMessage_CW, this, [this]() {
    Timer1->start();
  });
  connect(qnode, &QNode::receivedMessage_CCW, this, [this]() {
    Timer2->start();
  });
  connect(Timer1, &QTimer::timeout, this, &MainWindow::rotateJoint_CW);
  connect(Timer2, &QTimer::timeout, this, &MainWindow::rotateJoint_CCW);
}
```

Mainwindow의 소스파일 중 부분입니다. 팔을 회전시킬 때 사용할 타이머 1, 2를 초기화하고 타이머의 시작을 슬롯 함수로, 시계/반시계 방향 회전에 대한 수신을 시그널로 두어 묶었습니다. 또한 타임아웃을 시그널로, 회전 함수를 슬롯 함수로 묶어 팔의 회전을 시각적으로 나타냈습니다.

팔을 그리는 함수입니다. 이전 과제 부분을 가져와 구성했습니다.

```
53          QPainter painter(this);
54              painter.setRenderHint(QPainter::Antialiasing);
55
56              QPen pen1(Qt::black, 10);
57              painter.setPen(pen1);
58              painter.translate(260, 320);        //화면 중간쯤
59              painter.rotate(joint_degree);       // 관절 1 회전
60              painter.drawLine(0, 0, 100, 0);     //팔 그리기
61
62              // 2번째 관절
63              QPen pen2(Qt::red, 7);
64              painter.setPen(pen2);
65              painter.translate(100, 0);          // 첫 번째 팔 끝
66              painter.rotate(joint_degree);       // 관절 2 회전
67              painter.drawLine(0, 0, 80, 0);
68
69              QPen pen3(Qt::blue, 4);
70              painter.setPen(pen3);
71              painter.translate(80, 0);
72              painter.rotate(joint_degree);
73              painter.drawLine(0, 0, 50, 0);
74
75              QPen pen4(Qt::green, 4); //그리퍼 부분 표시
76              painter.setPen(pen4);
77              painter.translate(50,0);
78
79              painter.drawLine(0, 10, 0, 0);
80              painter.translate(0, 10);
81              painter.drawLine(0, 0, 7, 0);
82              painter.translate(0, -15);
83
84              painter.drawLine(0, 0, 0, 10);
85              painter.translate(0, -5);
86              painter.drawLine(0, 0, 7, 0);
```

```
90  void MainWindow::rotateJoint_CW()
91  {
92      joint_degree += 5;              // 각도 증가
93      if (joint_degree >= 356) joint_degree = 0;
94      repaint();
95  }
96
97  void MainWindow::rotateJoint_CCW()
98  {
99      joint_degree -= 5;              // 각도 감소
100     if (joint_degree <= 0) joint_degree = 356;
101     repaint();
102 }
```

마찬가지로 시계/반시계방향 회전 함수입니다. 타임아웃 시마다 호출되어 연속적인 움직임을 나타냅니다.

```
117 void MainWindow::on_pushButton_clicked()
118 {
119
120     QString text = ui->textEdit->toPlainText();
121     // qnode->setMessage(text);
122     qnode->publishing(text);
123     ui->label->setText(text);
124
125 }
126
127 void MainWindow::on_pushButton_2_clicked()
128 {
129     Timer1->stop();
130     Timer2->stop();
131     ui->label_2->setText("STOP");
132 }
```

Publishing 버튼과 stop버튼입니다. 메시지를 발행할 시 textEdit의 문자열을 읽어와 발행 함수를 호출합니다.

(2) qnode

```
21  QNode::QNode()
22  {
23    int count=0;
24    int argc = 0;
25    char** argv = NULL;
26    rclcpp::init(argc, argv);
27    node = std::make_shared<rclcpp::Node>("qnode");
28    publisher = node->create_publisher<std_msgs::msg::String>("topicname", 10);
29    subscriber = node->create_subscription<std_msgs::msg::String>(
30                  "topicname",
31                  10,
32                  std::bind(&QNode::topic_callback, this, std::placeholders::_1));
33  }
```

위 사진은 Qnode 소스파일에서의 생성자 정의 부분입니다.

```
56  void QNode::topic_callback(const std_msgs::msg::String::SharedPtr msg) {
57      qDebug() <<"Received in QNode:" << QString::fromStdString(msg->data);
58      if(msg->data=="CW"||msg->data=="cw"){
59        emit receivedMessage_CW(QString::fromStdString(msg->data));
60      }
61      else if(msg->data=="CCW"||msg->data=="ccw"){
62        emit receivedMessage_CCW(QString::fromStdString(msg->data));
63      }
64      else;
65  }
66
67  void QNode::publishing(QString text){
68    auto msg = std_msgs::msg::String();
69    msg.data = text.toStdString();
70    publisher->publish(msg);
71    RCLCPP_INFO(node->get_logger(), "Published message: '%s'", msg.data.c_str());
72  }
```

메시지 수신 함수와 발행 함수입니다. 수신함수에서는 메시지를 수신하고 나서 문자열에 따라 시계/반시계 방향의 회전 함수로 이어지는 시그널을 방출합니다.

3. 실행 화면

실행화면입니다. publishing 버튼을 누르면 textedit박스의 문자열이 자동으로 발행되고, 이를 수신한 뒤 로봇팔이 자동으로 움직입니다. stop버튼을 누르면 타이머가 멈추며 정지합니다.