Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Московский государственный технический университет
имени Н.Э. Баумана»
(МГТУ им. Н.Э. Баумана)

Факультет: Информатика и системы управления
Кафедра: Теоретическая информатика и компьютерные технологии

# Лабораторная работа №13
# «Изучение протокола MQTT»
# по курсу: «Языки и методы программирования»

Выполнил:
Студент группы ИУ9-22Б
Гнатенко Т. А.

Проверил:
Посевин Д. П.

Москва, 2022

# Цели

Целью лабораторной работы является приобретение навыка разработки на языке JAVA приложения выполняющего запись данных в топик публичного MQTT брокера и приложения чтения данных из топика.

# Задачи

Реализовать на языке JAVA запись и чтение данных в(из) топика в соответствии со своим вариантом.

# Решение

## Исходный код

**Get.java**

```
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;
import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
import java.util.Scanner;

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        String topic        = "IU/9";
        int qos             = 2;
        String content;
        String broker       = "tcp://mqtt.eclipseprojects.io:1883";
        String clientId     = "JavaSample";
        MemoryPersistence persistence = new MemoryPersistence();

        try {
            MqttClient sampleClient = new MqttClient(broker, clientId,
↪   persistence);
            MqttConnectOptions connOpts = new MqttConnectOptions();

            connOpts.setCleanSession(true);
            System.out.println("Connecting to broker: "+broker);
            sampleClient.connect(connOpts);
            System.out.println("Connected");

            Scanner in = new Scanner(System.in);
            while (in.hasNext()){
                content = in.nextLine();
                System.out.println("Publishing message: "+content);
                MqttMessage message = new
↪   MqttMessage(content.getBytes());
                message.setQos(qos);
                sampleClient.publish(topic, message);
                System.out.println("Message published");
            }


            sampleClient.disconnect();
```

```
            System.out.println("Disconnected");
            System.exit(0);
        } catch(MqttException me) {
            System.out.println("reason "+me.getReasonCode());
            System.out.println("msg "+me.getMessage());
            System.out.println("loc "+me.getLocalizedMessage());
            System.out.println("cause "+me.getCause());
            System.out.println("excep "+me);
            me.printStackTrace();
        }
    }
}
```

**Recieve.java**

```
import org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;
import org.eclipse.paho.client.mqttv3.MqttCallback;
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;

import java.sql.Struct;
import java.util.Scanner;

public class Main implements MqttCallback {

    MqttClient client;
    String topic = "IU/9";
    String broker = "tcp://mqtt.eclipseprojects.io:1883";

    public static void main(String[] args) {
        new Main().doDemo();
    }

    public void doDemo() {
        try {
            client = new MqttClient(broker, "Receiving");
            client.connect();
            client.setCallback(this);
            client.subscribe(topic);
            MqttMessage message = new MqttMessage();
        } catch (MqttException e) {
            e.printStackTrace();
        }
    }

    @Override
    public void connectionLost(Throwable cause) {
        // TODO Auto-generated method stub

    }

    @Override
    public void messageArrived(String topic, MqttMessage message) throws
     ↪  Exception {
        try {
            String str = message.toString();
            Scanner in = new Scanner(str);
            int[] arr = new int[6];
            for (int i = 0; i < 6; i++) {
                if (!in.hasNextInt()) {
                    throw new Exception("Incorrect data");
                }
```

```
                int n = in.nextInt();
                arr[i] = n;
            }
            if (arr[4] - Math.sqrt(Math.pow(arr[0] - arr[2], 2) +
             ↪  Math.pow(arr[1] - arr[3], 2)) >= arr[5]) {
                System.out.println(1);
            } else {
                System.out.println(0);
            }
            in.close();
        }catch (Exception e){
            System.out.println(e.getMessage());
        }
        System.out.println(message);
    }

    @Override
    public void deliveryComplete(IMqttDeliveryToken token) {
        // TODO Auto-generated method stub

    }

}
```