



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ _____ «Информатика и системы управления»

КАФЕДРА _____ «Теоретическая информатика и компьютерные технологии»

Летучка №3
по курсу «Языки и методы программирования»
«Изучение протокола MQTT»

Студент группы ИУ9-22Б Лавров Р. Д.

Преподаватель Посевин Д. П.

Москва 2025

1 Задание

Часть 1: Реализовать на языке JAVA запись и чтение данных в(из) топика в соответствии со своим вариантом, подготовить и прислать отчет, демонстрацию работы программы необходимо записать.

Часть 2: Реализовать два пользовательских GUI приложения с использованием библиотеки Swing: первое приложение должно содержать форму ввода исходных данных соответствующих вашему заданию, второе окно вывода — вычисление из вашего задания.

Часть 3: Реализовать пользовательское GUI приложение, которое одновременно выполняет функцию сендера и сабскрайбера. Это приложение должно содержать форму отправки текстового сообщения в топик /iu9/mqttchat и одновременно читать все сообщения, которые посылаются сругими сендерами.

2 Результаты

Часть 2

Листинг 1 — Реализация класса Chat

```
1 package org.example;
2 import javax.swing.*.*;
3 import java.awt.*.*;
4 public class Chat {
5     private JSpinner spinner1, spinner2, spinner3, spinner4, spinner5, spinner6;
6     private JTextArea textArea1; private JPanel mainPanel;
7     private JButton senderButton; private JLabel statusLabel;
8     private Sender sender; private Reader reader;
9     public Chat() {
10         senderButton.addActionListener(e -> {
11             try {
12                 int x1 = (int) spinner1.getValue(); int y1 = (int) spinner2.getValue();
13                 int z1 = (int) spinner3.getValue();
14                 int x2 = (int) spinner4.getValue(); int y2 = (int) spinner5.getValue();
15                 int z2 = (int) spinner6.getValue();
16                 sender.setNewCoordinates(x1, y1, z1, x2, y2, z2);
17                 statusLabel.setText("COOOlll");
18             } catch (Exception ex) {
19                 statusLabel.setText("Error: " + ex.getMessage());
20                 appendToTextArea("Error sending coordinates: " + ex.getMessage() + "\n");
21             }
22         });
23     }
24 }
```

Листинг 2 — Реализация класса Chat (продолжение)

```
1 public void appendToTextArea(String text) {
2     SwingUtilities.invokeLater(() -> {
3         textArea1.append(text);
4         textArea1.setCaretPosition(textArea1.getDocument().getLength());
5     });
6 }
7 public static void main(String[] args) {
8     SwingUtilities.invokeLater(() -> {
9         Chat chat = new Chat();
10
11         // Initialize MQTT components
12         chat.reader = new Reader(chat);
13         chat.sender = new Sender();
14
15         // Setup GUI frame
16         JFrame frame = new JFrame("3D Coordinates Sender");
17         frame.setContentPane(chat.mainPanel);
18         frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
19         frame.setSize(600, 500);
20         frame.setLocationRelativeTo(null);
21
22         new Thread(chat.reader::RunReader).start();
23         new Thread(chat.sender::RunSender).start();
24
25         frame.setVisible(true);
26     });
27 }
28 }
```

Листинг 3 — Реализация класса Reader

```
1 package org.example;
2 import org.eclipse.paho.client.mqttv3.*;
3 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
4
5 public class Reader implements Runnable {
6     private MqttClient client;
7     private boolean running = true;
8     private Chat chat;
9     public Reader(Chat chat) {this.chat = chat;}
10
11     public void RunReader() {
12         String broker = "tcp://broker.emqx.io:1883";
13         String topic = "/pstgu/geometry";
14         String clientId = "Reader_" + System.currentTimeMillis(); // Уникальный ID
15         try {
16             client = new MqttClient(broker, clientId, new MemoryPersistence());
17             MqttConnectOptions options = new MqttConnectOptions();
18             options.setCleanSession(true);
19             options.setAutomaticReconnect(true);
20             options.setConnectionTimeout(10);
21             options.setKeepAliveInterval(20);
```

Листинг 4 — Реализация класса Reader (продолжение)

```

1      client.setCallback(new MqttCallback() {
2          @Override
3          public void connectionLost(Throwable cause) {
4              System.out.println("Connection lost: " + cause.getMessage());
5          }
6          @Override
7          public void messageArrived(String topic, MqttMessage message) {
8              System.out.println("Message received on topic: " + topic);
9              String payload = new String(message.getPayload());
10             System.out.println("Raw message: " + payload);
11             String[] coords = payload.split(" ");
12             if (coords.length == 6) {
13                 try {
14                     int x1 = Integer.parseInt(coords[0]);
15                     int y1 = Integer.parseInt(coords[1]);
16                     int z1 = Integer.parseInt(coords[2]);
17                     int x2 = Integer.parseInt(coords[3]);
18                     int y2 = Integer.parseInt(coords[4]);
19                     int z2 = Integer.parseInt(coords[5]);
20                     double len1 = Math.sqrt(x1*x1 + y1*y1 + z1*z1);
21                     double len2 = Math.sqrt(x2*x2 + y2*y2 + z2*z2);
22                     double scl = x1*x2 + y1*y2 + z1*z2;
23                     String result = (scl == 0)
24                         ? String.format("Векторы ортогональные\nA: (%d,%d,%d)
25 \nB: (%d,%d,%d)\n",
26                             x1,y1,z1, x2,y2,z2)
27                         : String.format("Векторы неортогональные\nA: (%d,%d,%
28 d)\nB: (%d,%d,%d)\n",
29                             x1,y1,z1, x2,y2,z2);
30
31                     if (chat != null) {chat.appendTextArea(result);}
32                     System.out.println(result);
33
34                 } catch (NumberFormatException e) {
35                     String error = "Ошибка формата чисел: " + payload;
36                     System.err.println(error);
37                     if (chat != null) chat.appendTextArea(error);
38                 }
39             } else {
40                 String error = "Неверное количество координат: " + payload;
41                 System.err.println(error);
42                 if (chat != null) chat.appendTextArea(error);
43             }
44         }
45         @Override
46         public void deliveryComplete(IMqttDeliveryToken token) {
47             System.out.println("Delivery complete");
48         }
49     });
50     System.out.println("Connecting to broker: " + broker);
51     client.connect(options);
52     System.out.println("Subscribing to topic: " + topic);
53     client.subscribe(topic, 1);
54     System.out.println("Reader started successfully . Waiting for messages...");
55 } catch (MqttException e) {
56     System.err.println("Reader error: " + e.getMessage());
57     e.printStackTrace();
58 }
59 @Override
60 public void run() {RunReader();}

```

Листинг 5 — Реализация класса Sender

```
1 package org.example;
2 import org.eclipse.paho.client.mqttv3.*;
3 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
4
5 public class Sender implements Runnable {
6     private int x1 = 0, y1 = 0, z1 = 0;
7     private int x2 = 0, y2 = 0, z2 = 0;
8     private boolean send = false;
9     private boolean running = true;
10    private MqttClient client;
11
12    public synchronized void setNewCoordinates(int x1, int y1, int z1, int x2, int y2, int z2) {
13        this.x1 = x1;
14        this.y1 = y1;
15        this.z1 = z1;
16        this.x2 = x2;
17        this.y2 = y2;
18        this.z2 = z2;
19        this.send = true;
20    }
21
22    public void RunSender() {
23        String broker = "tcp://broker.emqx.io:1883";
24        String topic = "/pstgu/geometry";
25        String clientId = "TriangleSender_" + System.currentTimeMillis();
26
27        try {
28            client = new MqttClient(broker, clientId, new MemoryPersistence());
29
30            MqttConnectOptions options = new MqttConnectOptions();
31            options.setCleanSession(true);
32
33            client.connect(options);
34
35            while (running) {
36                try {
37                    if (send) {
38                        String res = String.format("%d %d %d %d %d %d", x1, y1, z1, x2, y2,
39z2);
40
41                        MqttMessage message = new MqttMessage(res.getBytes());
42                        message.setQos(1);
43                        client.publish(topic, message);
44                        System.out.println("Отправлено: " + res);
45                        send = false;
46                    }
47                    Thread.sleep(100);
48                } catch (InterruptedException e) {
49                    Thread.currentThread().interrupt();
50                    System.out.println("Sender прерван");
51                    break;
52                } catch (Exception e) {
53                    System.err.println("Ошибка отправки: " + e.getMessage());
54                }
55            }
56        } catch (MqttException e) {
57            System.err.println("Ошибка подключения Sender: " + e.getMessage());
58        }
59    }
60 }
```

Листинг 6 — Реализация класса Sender (продолжение)

```
1 public void stop() {
2     running = false;
3     try {
4         if (client != null && client.isConnected()) {
5             client.disconnect();
6             client.close();
7         }
8     } catch (MqttException e) {
9         System.err.println("Ошибка отключения Sender: " + e.getMessage());
10    }
11 }
12
13 @Override
14 public void run() {
15     RunSender();
16 }
17 }
```

Часть 2

Листинг 7 — Реализация класса Chat

```
1 package org.example;
2 import javax.swing.*;
3 import java.awt.*;
4
5 public class Chat {
6     private JTextArea textArea1;
7     private JPanel mainPanel;
8     private JButton senderButton;
9     private JLabel statusLabel;
10    private JTextField textField1;
11
12    private Sender sender;
13    private Reader reader;
14
15    public Chat() {
16        senderButton.addActionListener(e -> {
17            try {
18                sender.setMessage(textField1.getText());
19                statusLabel.setText("Сообщение отправлено");
20                textField1.setText("");
21            } catch (Exception ex) {
22                statusLabel.setText("Error: " + ex.getMessage());
23                appendToTextArea("Ошибка отправки сообщения: " + ex.getMessage() + "\n");
24            }
25        });
26    }
```

Листинг 8 — Реализация класса Chat (продолжение)

```
1 public void appendToTextArea(String text) {
2     SwingUtilities.invokeLater(() -> {
3         textArea1.append(text + "\n");
4         textArea1.setCaretPosition(textArea1.getDocument().getLength());
5     });
6 }
7 public static void main(String[] args) {
8     SwingUtilities.invokeLater(() -> {
9         Chat chat = new Chat();
10        chat.reader = new Reader(chat);
11        chat.sender = new Sender();
12        JFrame frame = new JFrame("BMSTU Chat");
13        frame.setContentPane(chat.mainPanel);
14        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
15        frame.setSize(600, 500);
16        frame.setLocationRelativeTo(null);
17        new Thread(chat.reader::RunReader).start();
18        new Thread(chat.sender::RunSender).start();
19        frame.setVisible(true);
20    });
21 }
22
23 {
24 }
```

Листинг 9 — Реализация класса Reader

```
1 package org.example;
2 import org.eclipse.paho.client.mqttv3.*;
3 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
4
5 public class Reader implements Runnable {
6     private MqttClient client;
7     private boolean running = true;
8     private Chat chat;
9     public String clientId = "veterok_reader";
10
11     public Reader(Chat chat) {
12         this.chat = chat;
13     }
14
15     public void RunReader() {
16         String broker = "tcp://broker.emqx.io:1883";
17         String topic = "/iu9/mqttchat";
18
19         try {
20             client = new MqttClient(broker, clientId, new MemoryPersistence());
21             MqttConnectOptions options = new MqttConnectOptions();
```

Листинг 10 — Реализация класса Reader (продолжение)

```

1      client .setCallback(new MqttCallback() {
2          @Override
3          public void connectionLost(Throwable cause) {
4              System.out.println("Соединение потеряно: " + cause.getMessage());
5              reconnect();
6          }
7
8          @Override
9          public void messageArrived(String topic, MqttMessage message) {
10             String payload = new String(message.getPayload());
11             String[] parts = payload.split("\\|", 2);
12             if (chat != null) {
13                 chat.appendTextArea(parts[0] + ": " + parts[1]);
14             }
15         }
16
17         @Override
18         public void deliveryComplete(IMqttDeliveryToken token) {}
19     });
20
21     client .connect(options);
22     client .subscribe(topic, 1);
23     System.out.println("Подключено к брокеру. Ожидание сообщения ... ");
24
25     } catch (MqttException e) {
26         System.err.println("Ошибка подключения: " + e.getMessage());
27     }
28 }
29
30 private void reconnect() {
31     while (running) {
32         try {
33             Thread.sleep(5000);
34             if (! client .isConnected()) {
35                 client .reconnect();
36                 System.out.println("Переподключение успешно");
37                 return;
38             }
39         } catch (Exception e) {
40             System.err.println("Ошибка переподключения: " + e.getMessage());
41         }
42     }
43 }
44
45 public void stop() {
46     running = false;
47     try {
48         if (client != null && client.isConnected()) {
49             client .disconnect();
50             client .close();
51         }
52     } catch (MqttException e) {
53         System.err.println("Ошибка отключения: " + e.getMessage());
54     }
55 }
56
57 @Override
58 public void run() {
59     RunReader();
60 }
61 }
```


Листинг 11 — Реализация класса Sender

```

1 package org.example;
2 import org.eclipse.paho.client.mqttv3.*;
3 import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;
4
5 public class Sender implements Runnable {
6     private String message = "";
7     private boolean send = false;
8     private boolean running = true;
9     private MqttClient client;
10    public String clientId = "veterok";
11
12    public synchronized void setMessage(String message) {
13        this.message = message;
14        this.send = true;
15    }
16
17    public void RunSender() {
18        String broker = "tcp://broker.emqx.io:1883";
19        String topic = "/iu9/mqttchat";
20
21        try {
22            client = new MqttClient(broker, clientId, new MemoryPersistence());
23            MqttConnectOptions options = new MqttConnectOptions();
24            options.setCleanSession(true);
25            client.connect(options);
26
27            while (running) {
28                try {
29                    if (send) {
30                        String fullMessage = String.format("%s|%s", clientId, message);
31                        MqttMessage message = new MqttMessage(fullMessage.getBytes());
32                        message.setQos(1);
33                        client.publish(topic, message);
34                        System.out.println("Отправлено: " + this.message);
35                        synchronized (this) {
36                            send = false;
37                        }
38                    }
39                    Thread.sleep(100);
40                } catch (InterruptedException e) {
41                    Thread.currentThread().interrupt();
42                    System.out.println("Sender прерван");
43                    break;
44                } catch (Exception e) {
45                    System.err.println("Ошибка отправки: " + e.getMessage());
46                }
47            }
48        } catch (MqttException e) {
49            System.err.println("Ошибка подключения Sender: " + e.getMessage());
50        }
51    }
52
53    @Override
54    public void run() {
55        RunSender();
56    }
57 }

```

3D Coordinates Sender

Вектор 1

Вектор 2

0

4

6

3

0

-4

Sender

Векторы ортогональные

A: (0,0,6)

B: (0,0,0)

Векторы неортогональные

A: (0,4,6)

B: (3,0,-4)

COOOIII

BMSTU Chat

Sender

vov4ik: здесь кто-нибудь нет?

veterok: Есть

vov4ik: добрый вечер

veterok: Вечер добрый

vov4ik: как жизнь?

veterok: Плохо, ведь скоро коллок у Безверха

vov4ik: 😞

vov4ik: тяжела да неказиста жизнь простого программиста

veterok: Ну может за летучку норм накинута, тогда хорошо

veterok: Ахахаах

vov4ik: думаю заслуженно

veterok:)

veterok: У меня смайликов нет поэтому через скобки

vov4ik: эхх

vov4ik: увы

veterok: Как 2000х

vov4ik:))))))))))))

vov4ik: Мне пора. Удачи!

veterok: Ну ладно думаю пора спать

veterok: Отправлю ролик и пойду

vov4ik: спокойной ночи

veterok: И тебе

Сообщение отправлено

10

3 Вывод

Благодаря этой летучке, я смог написать чат используя mqtt