MQTT Binary Stream Client

Author: Yizhou Lu

Develop a Java program that creates a raw MQTT CONNECT packet and sends it to an MQTT broker using a TCP socket. Then, the program will read the CONNACK packet received from the broker and print its contents.

Code Sections

1. Connecting to MQTT Broker

The initial section establishes a TCP connection to the MQTT broker running on the localhost at port 1883.

```
Socket socket = new Socket("localhost", 1883);
InputStream in = socket.getInputStream();
OutputStream out = socket.getOutputStream();
```

2. Sending CONNECT Packet

The CONNECT packet is crafted as per the MQTT protocol specifications. It includes details such as protocol name, protocol level, connect flags, client ID, etc.

```
byte[] connectPacket = {
    // MQTT CONNECT packet details
};

// Calculate the Remaining Length and update the packet
int remainingLength = connectPacket.length - 2;
connectPacket[1] = (byte) remainingLength;

out.write(connectPacket); // Send CONNECT packet to Broker
```

3. Receiving CONNACK

After sending the CONNECT packet, the program reads the CONNACK response from the broker.

```
byte[] connackPacket = new byte[4];
in.read(connackPacket);

// Display the contents of the CONNACK message
```

4. Sending PUBLISH Packet

Following a successful connection, a PUBLISH packet is created and sent to the broker.

```
byte[] publishPacket = {
    // MQTT PUBLISH packet details
};

// Calculate the Remaining Length and update the packet
int remainingLength2 = publishPacket.length - 2;
publishPacket[1] = (byte) remainingLength2;

out.write(publishPacket); // Send PUBLISH packet to Broker
```

5. Receiving PUBACK

After sending the PUBLISH packet, the program reads the PUBACK response from the broker.

```
byte[] pubackPacket = new byte[4];
in.read(pubackPacket);

// Display the contents of the PUBACK message
```

6. Closing Connection

Finally, the TCP connection is closed.

```
socket.close();
```

Complete Code

```
import java.io.InputStream;
import java.io.OutputStream;
import java.net.Socket;
import java.io.IOException;

public class MQTTBinaryClient {

    public static void main(String[] args) {
        try{
            Socket socket = new Socket("localhost", 1883); // Broker adress
            InputStream in = socket.getInputStream();
            OutputStream out = socket.getOutputStream();

            // CONNECT Packet
            byte[] connectPacket = {
                0x10, // Comman type: 0001  Control flags: 0000
                0x13, // Remaining length: 0001 0011 (19)
                0x00, 0x04, // Protocol name length - 4
```

```
                'M', 'Q', 'T', 'T', // Protocol name - MQTT
                0x04, // Protocol level - MQTT protocol version is 4
                0x02, // Connect flag: 0000 0010, the second bit represents Clean
Session, which is true here
                0x00, 0x3c, // Keep alive timer: 0000 0000 0011 1100 - 60 Sec
                0x00, 0x03, // Client ID length - 3
                'A', 'd', 'i' // Client ID - Adi
            };

            // Calculate the Remaining Length
            int remainingLength = connectPacket.length - 2;
            connectPacket[1] = (byte) remainingLength;

            out.write(connectPacket); // Send CONNECT packet to Broker

            // Read CONNACK
            byte[] connackPacket = new byte[4];
            in.read(connackPacket);

            // Display the contents of the CONNACK message
            System.out.println("Received CONNACK byte: ");
            for (byte b : connackPacket) {
                System.out.printf("0x%02x ", b);
            }

            System.out.println();

            // PUBLISH Packet
            byte[] publishPacket = {
                0x32, // Command type: PUBLISH  Control flag: 1101 (DUP: 1, QoS:
2, Retain: 1)
                0x0a, // Remaining length: 10
                0x00, 0x07, // Topic name length
                'm', 'q', 't', 't', 'l', 'a', 'b', // Topic name: mqttlab
                0x48, 0x65, 0x79, 0x42, 'r', 'o' // Payload: HeyBro
            };

            // Calculate the Remaining Length
            int remainingLength2 = publishPacket.length - 2;
            publishPacket[1] = (byte) remainingLength2;

            out.write(publishPacket); // Send PUBLISH packet to Broker

            // Read PUBACK
            byte[] pubackPacket = new byte[4];
            in.read(pubackPacket);

            // Display the contents of the PUBACK message
            System.out.println("Received PUBACK byte: ");
            for (byte b : pubackPacket) {
                System.out.printf("0x%02x ", b);
            }

            socket.close();
```

```
            } catch (IOException e){
                e.printStackTrace();
            }
        }
    }
```