

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

www.dei.estg.ipleiria.pt

Curso Técnico Superior Profissional Programação de Sistemas de Informação

Serviços e Interoperabilidade de Sistemas

MESSAGING

Objetivos:

– Implementar um exemplo de *messaging* com *Mosquitto*.

Duração: 1 aula

©2021-2022: {bruno .madeira, mario.fernandes, nuno.costa, nuno.costa, romeu.paz}@ipleiria.pt

Cofinanciado por:







Descrição

O Eclipse Mosquitto é um projeto open-source que implementa o protocolo MQTT 3.1 e 3.1.1. e poderá ser usado para ligar diferentes sistemas ou aplicações seguindo o modelo Publish/Subscribe.

Como executar:

- 1. Mosquitto Messeging
 - a. Descarregar Mosquitto:

https://mosquitto.org/download/

- Teste abrindo 3 consolas da linha de comandos na pasta de instalação
 (C:\Program Files\mosquitto, por exemplo):
 - i. Na primeira inicie o Mosquitto executando:

```
mosquitto -v
```

ii. Na segunda inicie uma subscrição do canal "Teste" (subscribe):

```
mosquitto sub -v -t "Teste" -h 127.0.0.1
```

iii. Na terceira faça uma publicação para o canal "Teste" (publish):

```
mosquitto_pub -t "Teste" -m "Minha mensagem de teste 1" -h
127.0.0.1
```

iv. Observe o resultado na segunda consola

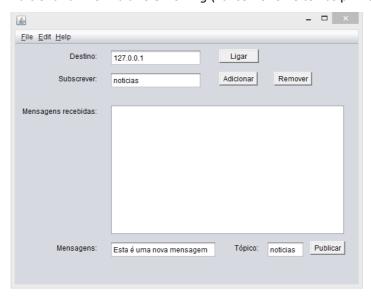
em:

- 2. Crie uma aplicação em JAVA (Netbeans) para publicar e subscrever:
 - a. Para isso é necessário utilizar a API para JAVA "Eclipse Paho Java Client".

Efectuar download do ficheiro

https://search.maven.org/search?q=a:org.eclipse.paho.client.mqttv3

- b. Criar um novo projecto de aplicação java no Netbeans
 - i. Adicionar um formulário em swing (Tal como foi feito nas primeiras aulas)



(Exemplo do formulário a criar)

ii. Implemente o código para o botão "Ligar"

```
private void jButtonLigarActionPerformed(java.awt.event.ActionEvent evt) {
         try{
                   String clientID= "Teste";
                   myClient = new MqttClient("tcp://" + jTextFieldDestino.getText() + ":1883", clientID, null);
                   myClient.setCallback(new MqttCallback() {
                             @Override
                            public void messageArrived(String topic, MqttMessage msg) throws Exception {
                                      String messageBody = new String(msg.getPayload());
                                      jTextAreaMsg.setText(messageBody + "\r\n" + jTextAreaMsg.getText());
                            @Override
                            public void deliveryComplete(IMqttDeliveryToken token) {
                                      jTextAreaMsg.setText("mensagem enviada" + "\r\n" + jTextAreaMsg.getText());
                            @Override
                            public void connectionLost(Throwable exception) {
                                      jTextAreaMsg.setText("Impossível enviar mensagem" + "\r\n" + jTextAreaMsg.getText());
                   });
                   myClient.connect();
                   ¡TextAreaMsg.setText("Ligado ao Mosquitto" + "\r\n" + ¡TextAreaMsg.getText());
         }
         catch(MqttException ex)
                   jTextAreaMsg.setText("Impossível ligar ao Mosquitto" + "\r\n" + jTextAreaMsg.getText());
         }
}
```

iii. Implemente o código para os botões "Adicionar" e "Remover"

```
private void jButtonAdicionarActionPerformed(java.awt.event.ActionEvent evt) {
         try {
                   String topico = jTextFieldSubscrever.getText();
                   int subQoS= 0;
                   myClient.subscribe(topico, subQoS);
                   jTextAreaMsg.setText("subscreveu " + topico + "\r\n" + jTextAreaMsg.getText());
         } catch (Exception e) {
                   jTextAreaMsg.setText("Impossível subscrever" + "\r\n" + jTextAreaMsg.getText());
         }
}
private void jButtonRemoverActionPerformed(java.awt.event.ActionEvent evt) {
         try {
                   String[] arrTopicos = new String[1];
                   arrTopicos[0] = jTextFieldSubscrever.getText();
                   myClient.unsubscribe(arrTopicos);
                   jTextAreaMsg.setText("removeu subscrição " + arrTopicos[0] + "\r\n" + jTextAreaMsg.getText());
         } catch (Exception e) {
                   jTextAreaMsg.setText("Impossível remover subscrição" + "\r\n" + jTextAreaMsg.getText());
         }
}
```

iv. Implemente o código para o botão "Publicar"

```
private void jButtonPublicarActionPerformed(java.awt.event.ActionEvent evt) {
    String topico = jTextFieldTopico.getText();
    MqttTopic noticiasTopic= myClient.getTopic(topico);

String pubMsg = jTextFieldMensagem.getText();
    int pubQoS= 0;

MqttMessage message = new MqttMessage(pubMsg.getBytes());

message.setQos(pubQoS);
    message.setRetained(false);

try {
        noticiasTopic.publish(message);
    } catch (Exception e) {
            jTextAreaMsg.setText("Impossível enviar mensagem" + "\r\n" + jTextAreaMsg.getText());
    }
}
```

- v. Altere a linha "message.setRetained(false);" para "true"
 - 1. Ligue ao mosquito mas não subscreva nada.
 - 2. Publique uma mensagem para o tópico "noticias"
 - 3. Subescreva agora o tópico "noticias"
 - 4. Comente os resultados
- vi. Utilize a linha de comandos para mandar uma mensagem e recebê-la na aplicação Java
- vii. Utilize a Aplicação Java para mandar mensagens para a máquina de um colega (Atenção: Firewall do Windows: abrir porta: 1883)