ĐẠI HỌC QUỐC GIA HÀ NỘI
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO BÀI TẬP LỚN MÔN LẬP TRÌNH MẠNG

ĐỀ BÀI: Xây dựng chương trình kiểu publish/subscribe như MQTT cho các thiết bị cảm biến

Sinh viên:

- Hoàng Minh Đức Anh - 18020003
- Nguyễn Chí Thành - 18020053
- Nguyễn Hải Long - 18020037

Hà Nội, tháng 12 năm 2021

MỤC LỤC

1. Yêu cầu bài toán

I. Chương_trình_hiển_thị_thông_tin_cảm_biến_A kết nối với Broker, đăng ký nhận thông tin tại topic /locationA/sensorA (locationA, sensorA: tham số có thể tuỳ biến)

II. Chương_trình_sinh_dữ_liệu_cảm_biến_A kết nối với Broker, tự động sinh dữ liệu cảm biến và publish dữ liệu lên Broker vào topic /locationA/sensorA

III. Chương_trình_hiển_thị_thông_tin_cảm_biến_A nhận dữ liệu được publish vào topic /locationA/sensorA tức dữ liệu của cảm biến A và hiển thị thông tin cho người dùng

IV. Nhiều chương trình sinh dữ liệu cảm biến cùng kết nối và publish dữ liệu lên Broker với các topic khác nhau

V. Nhiều chương trình sinh dữ liệu cảm biến cùng kết nối vào Broker và mỗi chương trình nhận và hiển thị các thông tin từ một topic khác nhau theo thời gian thực

VI. (thêm) Thi công vẽ UI để nhìn chương trình dễ dàng và đẹp hơn.

2. Quá trình làm việc

- Họp team, đọc kỹ đề bài, phân tích đặc tả yêu cầu, từ đó chia việc cho từng thành viên.
- Phân chia công việc:
    + Thành: Đảm nhiệm thiết kế Protocol và Broker
    + Long: Đảm nhiệm Publisher
    + Đức Anh: Đảm nhiệm Subscriber
- Timeline:
    + 27/11: Họp team, phân chia công việc. Chốt Protocol. Bắt đầu làm Broker.
    + 01/12: Hoàn thành phần lõi của Broker.
    + 02/12: Bắt đầu làm Publisher và Subscriber.
    + 10/12: Hoàn thành phần lõi của Publisher (gửi, nhận, sinh thông báo) và Subscriber (Sử dụng giao diện dòng lệnh).
    + 11/12: Sửa một vài lỗi nhỏ của Publisher và Broker. Bắt đầu làm GUI cho Subscriber.
    + 12/12: Hoàn thành cửa sổ hiển thị Output từ Server của Subscriber.
    + 13/12: Hoàn thành cửa sổ nhập Command, cửa sổ nhập IP + Port để kết nối cho Subscriber
    + 14/12: Publisher có thể sinh ngẫu nhiên nhiều topic hơn. Bắt đầu viết báo cáo
    + 16/12: Hoàn thiện báo cáo.

3. Mô tả giao thức

Quá trình gửi nhận tin nhắn sẽ gồm 3 thành phần tham gia:

- Subscriber: Đóng vai trò là Client đăng ký nhận tin nhắn từ 1 hoặc nhiều topic.
- Publisher: Đóng vai trò là Client tạo ra các tin nhắn và gửi tới Broker
- Broker: Đóng vai trò là Server tiếp nhận tin nhắn từ Publisher, sau đó gửi tới Subscriber dựa theo topic của bản tin.

Các câu lệnh trong giao thức:

- CONNECT: Client cần gửi lệnh này sau khi tạo kết nối tới Server. Nếu kết nối được chấp nhận, Server trả về CONNACK [ClientId].
- SUBSCRIBE [Topic]: Client đăng ký nhận thông báo từ Topic. Nếu đăng ký được chấp nhận, Server trả về SUBACK [Topic].
- UNSUBSCRIBE [Topic]: Client huỷ đăng ký nhận tin từ Topic. Nếu huỷ đăng ký thành công, Server trả về UNSUBACK [Topic].
- PSUBSCRIBE [Pattern]: Client đăng ký nhận thông báo từ Topic phù hợp với Pattern. Ví dụ, khi Client PSUBSCRIBE /LocationA/ thì sẽ nhận được từ /LocationA/Sensor1, /LocationA/Sensor2, … Nếu đăng ký được chấp nhận, Server trả về PSUBACK [Pattern].
- PUSUBSCRIBE [Pattern]: Client hủy đăng ký nhận thông báo từ Topic phù hợp với Pattern. Nếu huỷ đăng ký thành công, Server trả về PUNSUBACK [Pattern].
- PUBLISH [Topic] [Message]: Client gửi Message tới Topic. Nếu gửi thành công, Server sẽ trả về PUBACK. Sau khi nhận được lệnh PUBLISH, Server sẽ gửi Message tới các Client đã đăng ký Topic theo dạng: PUBLISH [Topic] [Message].

4. Chi tiết về chương trình

Ngôn ngữ chính : Java.Thư viện hỗ trợ thêm: Javafx cho phần GUI của Subscribe.

4.1. Mô tả hoạt động của Server:

- Khi khởi động sẽ sinh ra 2 thread chính
  + 1 thread để lắng nghe kết nối của Client
  + 1 thread để gửi thông báo PUBLISH tới Client
- Ở luồng thứ nhất, khi có kết nối tới từ Client, Server sẽ sinh ra 1 thread mới để tiếp nhận và xử lý các tin nhắn tới:
  + Khi nhận được CONNECT, Server sinh ra 1 UUID và dùng nó để định danh client

+ Khi nhận được SUBSCRIBE, Server thêm ClientId vào danh sách đăng ký topic đó. Sử dụng ConcurrentHashMap có Key là tên topic và Value là 1 Set các ClientId đăng ký topic đó.
+ Khi nhận được UNSUBSCRIBE, Server sẽ xoá Client đó ra khỏi danh sách đăng ký.
+ PSUBSCRIBE và PUNSUBSCRIBE tương tự như SUBSCRIBE và UNSUBSCRIBE.
+ Khi nhận được PUBLISH, Server sẽ thêm message vào message queue để đợi được gửi tới các Client đã đăng ký. Message queue sử dụng kiểu dữ liệu là BlockingQueue.
+ Khi Client ngắt kết nối, Server sẽ xóa Client khỏi danh sách đang hoạt động và danh sách đăng ký của các topic đã đăng ký, sau đó dừng thread.

- Ở luồng thứ hai, Server sẽ chạy một vòng lặp liên tục kiểm tra message queue xem có tin nhắn cần gửi không, nếu có thì lấy danh sách các Client đã đăng ký topic đó và gửi tin nhắn tới Client. Với các Pattern, Server cần duyệt tất cả các Pattern đang được đăng ký để kiểm tra, nếu Topic match với Pattern thì thêm những người đang đăng ký Pattern đó vào danh sách gửi tin.

4.2. Mô tả hoạt động của Publisher:

Nhiệm vụ của Publisher ở trong bài tập lớn lần này là tự sinh ra các câu thông báo và gửi nó đến Server. 2 công việc trên sẽ được tách riêng ra 2 thread để không ảnh hưởng đến nhau: MessageGenerator và ServerInteractHandler.

Hai luồng sẽ thực hiện thao tác chung trên một queue, nên để tránh gây tắc nghẽn, chương trình đã sử dụng BlockingQueue vì nó có thể triển khai dễ dàng và an toàn chia sẻ giữa các thread. BlockingQueue này sẽ được khởi tạo ở Class Publisher và truyền cho 2 thread. Nếu đúng logic thì ở Publisher sẽ tạo 2 thread riêng biệt, nhưng trong bài này nhóm em đã thảo luận để thống nhất chung là chỉ khi kết nối được server thì mới bắt đầu tạo thông báo, nên luồng MessageGenerator đã được chuyển vào khởi chạy trong thread ServerInteractHandler. Chi tiết về 2 luồng như sau:

- Luồng tạo thông báo: Class tạo thông báo chấp nhận 2 tham số khởi tạo là queue chung và định nghĩa của tín hiệu hết thông báo. sinh ra dữ liệu với hàm generateMessage(), đẩy vào queue. Hàm này sẽ sinh ra thông báo random topic và random câu thông báo. Trong chương trình thì có thực hiện delay để tránh sinh quá nhiều dẫn đến gửi quá nhiều request đến server.
- Luồng tương tác với Server: Class tiếp nhận các tham số khởi tạo bao gồm địa chỉ server, cổng server, queue và định nghĩa của tín hiệu hết thông báo. Sau đó từ các tham số này

class này sẽ kết nối với Server. Khi kết nối thành công theo protocol thì sẽ bắt đầu tạo thread sinh tin nhắn. Bằng việc sử dụng hàm take() thì thread này sẽ chờ cho đến khi queue có các thông báo. Ở đây có sử dụng một trường đánh dấu là lastSentSuccess. Nếu gửi dữ liệu thành công cho server sẽ nhận được "PUBACK" thì lastSentSuccess = true, lúc này ta tiến hành lấy thông báo tiếp theo. Còn trong trường hợp không nhận được "PUBACK" hoặc quá thời gian mà Server không giao tiếp gì thì tiến hành gửi lại thông báo cũ.

4.3. Mô tả hoạt động của Subscriber:

Subscriber có các chức năng đăng ký nhận thông tin từ Publisher thông qua Broker. Xây dựng Subscriber khá đơn giản với tính năng Gửi và Nhận thông báo thông qua cửa sổ giao diện người dùng. Sau khi kết nối tới Broker thì sinh ra thêm 1 thread có nhiệm vụ liên tục đọc thông tin từ Server gửi về để cập nhật vào cửa sổ Output. GUI của Subscriber được xây dựng bằng thư viện JavaFx.

5. Tổng kết

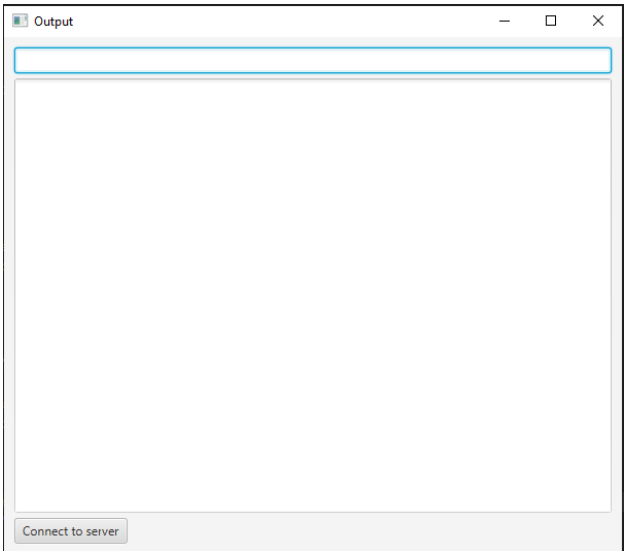
Source code: https://github.com/eruchii/NetworkProgramming-MQTT
Thiết lập server ở cổng 9999.
Ở phía Publisher : Nhập IP và Port. Ta sẽ thấy thông báo Done connect to server - thông báo connect thành công với server. Sau khi connect thành công thì Publisher bắt đầu gửi thông báo:

```
Nhap IP: localhost
Nhap Port: 9999
Done connect to server
Sending publish: PUBLISH /LocationA/Temperature The Temperature is 6
Sending publish: PUBLISH /LocationC/AirQuality The AirQuality is 75
Sending publish: PUBLISH /LocationA/AirQuality The AirQuality is 69
Sending publish: PUBLISH /LocationD/AirQuality The AirQuality is 43
```

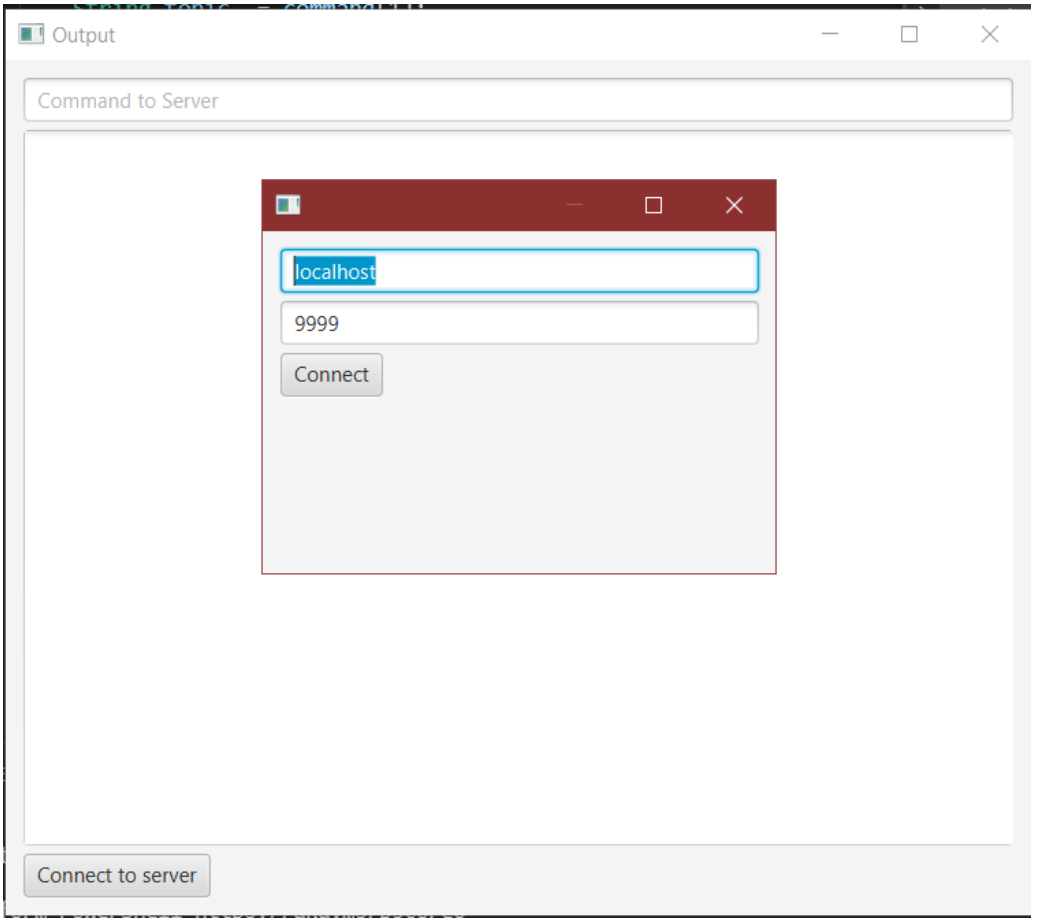
Phía Server cũng nhận được các thông báo publish:

```
Port: 9999
Server is waiting to accept user...
New connection: 127.0.0.1
CONNECT
ed74f3ce-18b8-4702-9f5b-7cc130277551
PUBLISH /LocationA/Temperature The Temperature is 6
PUBLISH /LocationC/AirQuality The AirQuality is 75
PUBLISH /LocationA/AirQuality The AirQuality is 69
PUBLISH /LocationD/AirQuality The AirQuality is 43
```

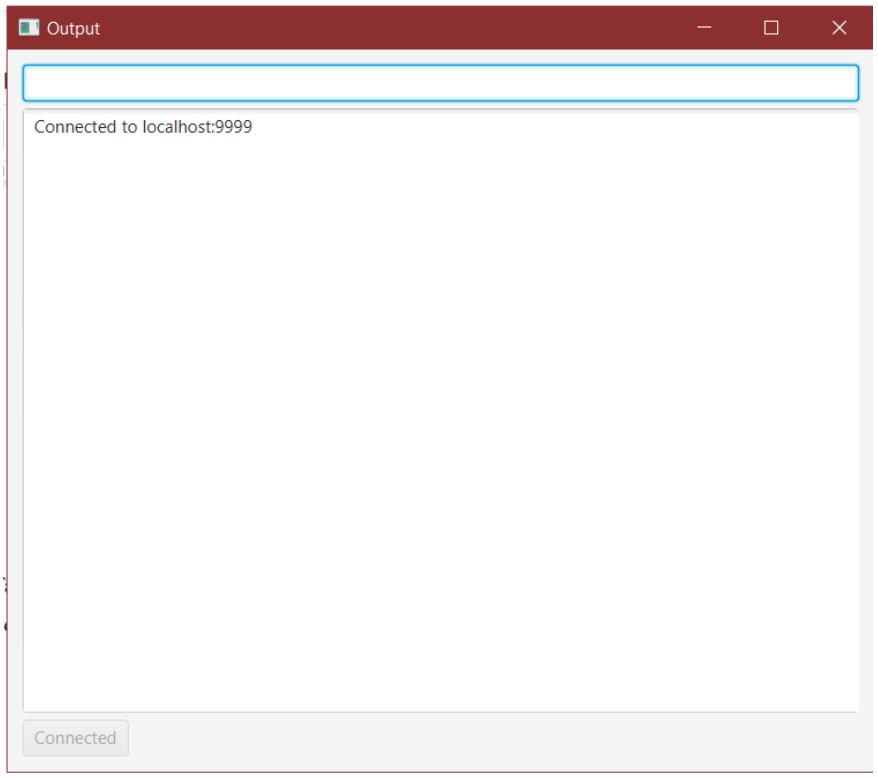
Giao diện Subscriber:



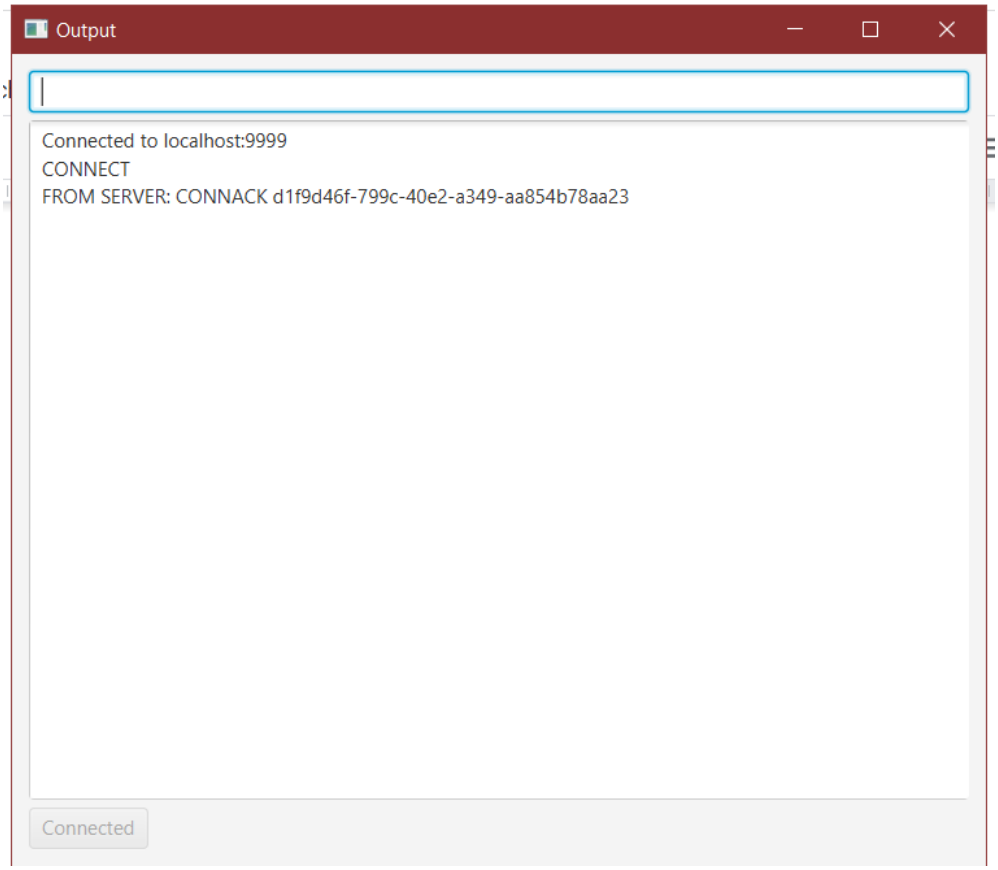
Nhấn nút Connect to server để nhập Hostname và Port:



Ta nhập IP và Port. Sau đó nhấn Connect. Nếu kết nối thành công sẽ hiển thị Connected to [host]:[port]:



Gõ CONNECT để đăng nhập vào Server. Server sẽ trả lại CONNACK [clientId]:



Sau khi đăng nhập thành công vào Server, người dùng bắt đầu tương tác bằng các lệnh SUBSCRIBE, UNSUBSCRIBE, PSUBSCRIBE, PUNSUBSCRIBE, PUBLISH.