**Kafka 스케일링**

**Kafka Partition vs. Consumers**

* Kafka Topic 생성시, default partition은 1개로 생성된다.
* kafka에서 하나의 Partition은 반드시 하나의 Consumer가 매칭되어 메시지를 소비한다.
* Partiton 수보다 동일한 Group id를 가진 Consumer 수가 많다면 일부 Consumer들은 partition에 binding되지 못해 message를 Polling 하지 못하는 현상이 일어난다.
* 아래의 Instruction을 따라 일부 Consumer가 메시지를 poll 해오지 못하는 현상을 확인한다.
* Order 서비스 시작

**cd order**

**mvn spring-boot:run**

* Product1 서비스 시작

**cd product1**

**mvn spring-boot:run**

* Product2 서비스 시작

**cd product2**

**mvn spring-boot:run**

Product1 서비스와는 달리 Product2 마이크로서비스의 Console 창을 통해 파티션 할당이 일어나지 않았음을 확인할 수 있다.  
partitions assigned: []

* 토픽정보와 Consumer 그룹정보를 확인한다.

**/usr/local/kafka/bin/kafka-topics.sh --bootstrap-server 127.0.0.1:9092 --topic kafkatest --describe**

**/usr/local/kafka/bin/kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group product**

Product Consumer가 2개임에도 파티션이 1개이므로, 매칭된 Consumer가 1개로 확인된다.

**Kafka Partition Scale out**

* Kafka Partition을 확장한다.

**$kafka\_home/bin/kafka-topics.sh --zookeeper localhost:2181 --alter --topic kafkatest -partitions 2**

* Product2 마이크로서비스를 재시작하거나 2~3분 정도 기다리면 Partition Rebalancing이 일어나면서 Product2 서비스도 partition assigned로 바뀌며 message를 Polling할 수 있는 상태로 변경된다.
* 토픽정보와 Consumer Group 정보를 재확인한다.

**/usr/local/kafka/bin/kafka-topics.sh --bootstrap-server 127.0.0.1:9092 --topic kafkatest --describe**

**/usr/local/kafka/bin/kafka-consumer-groups.sh --bootstrap-server localhost:9092 --describe --group product**

Partition 0,1 각각에 Consumer가 매핑된 것을 확인할 수 있다.

* Order 서비스에 POST로 메시지를 발행하면 Product 1, Product 2 서비스가 차례로 메시지를 수신한다.

**http POST :8081/orders message=1st-Order**

**http POST :8081/orders message=2nd-Order**

**http POST :8081/orders message=3rd-Order**

**http POST :8081/orders message=4th-Order**