



Kafka producer + consumer 실습

*window를 기준으로 한 글로 다른 운영체제에서는 다를 수 있으니 유의 부탁드립니다

1. 카프카 설치

<https://kafka.apache.org/downloads>

링크 클릭(컨트롤 클릭) -> 공식 다운로드 사이트 or 미러 사이트 에서 카프카 압축 파일 다운로드 -> 압축 해제

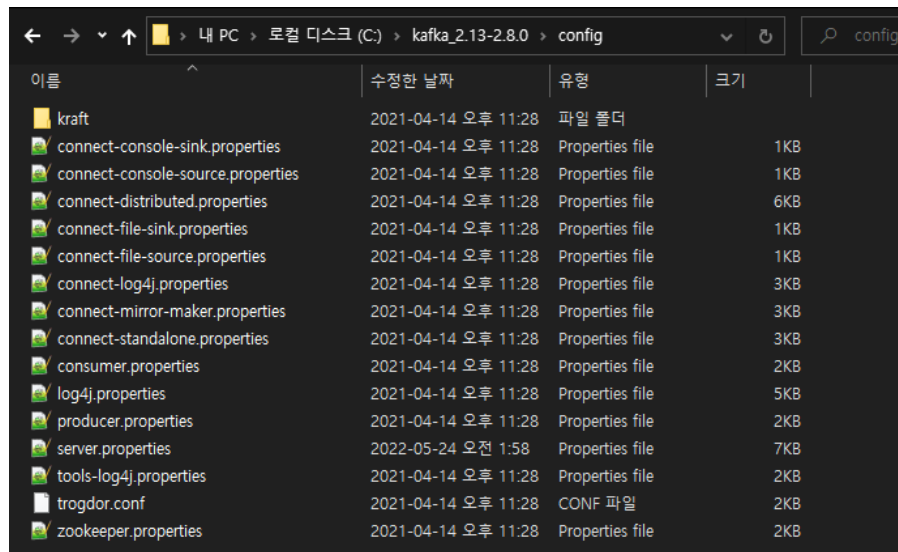
3.2.0

- Released May 17, 2022
- [Release Notes](#)
- Source download: [kafka-3.2.0-src.tgz](#) (asc, sha512)
- Binary downloads:
 - Scala 2.12 - [kafka_2.12-3.2.0.tgz](#) (asc, sha512)
 - Scala 2.13 - [kafka_2.13-3.2.0.tgz](#) (asc, sha512)

2. 카프카 설정

디렉토리 config로 이동 (ex. C:\kafka_2.13-2.8.0\config) ->

server.properties 파일 텍스트 편집기(ex. Notepad++)로 오픈 ->



log.dirs=/tmp/kafka-logs 를 카프카 설치 경로로 변경 (ex. log.dirs=C:\\kafka_2.13-2.8.0\\logs)

```
57 ##### Log Basics #####
58
59 # A comma separated list of directories under which to store log files
60 log.dirs=C:\\kafka_2.13-2.8.0\\logs
61
```

3. 카프카 실행

a. 주키퍼 실행

Zookeeper 다운로드 경로\bin에서 zkServer.cmd 클릭

주키퍼 설치 및 설정

b. 카프카 실행

주키퍼 cmd창 유지한 채로 새로운 cmd 실행 ->

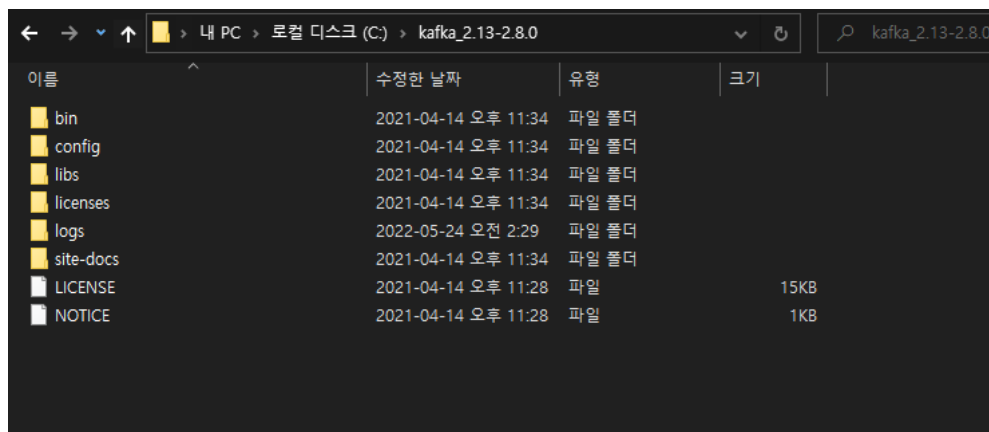
kafka 설치 경로로 이동(ex. cd C:\kafka_2.13-2.8.0)

bin\windows\kafka-server-start.bat config\server.properties 입력

```
명령 프롬프트 - bin\windows\kafka-server-start.bat config\server.properties
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\W82109>cd C:\kafka_2.13-2.8.0

C:\kafka_2.13-2.8.0>bin\windows\kafka-server-start.bat config\server.properties
[2022-05-24 02:28:52,208] INFO Registered kafka:type=kafka.Log4jController MBean (kafka.utils.Log4jControllerRegistration$)
[2022-05-24 02:28:52,721] INFO Setting -D jdk.tls.rejectClientInitiatedRenegotiation=true to disable client-initiated TLS renegotiation (org.apache.zookeeper.common.X509Util)
[2022-05-24 02:28:52,892] INFO starting (kafka.server.KafkaServer)
[2022-05-24 02:28:52,893] INFO Connecting to zookeeper on localhost:2181 (kafka.server.KafkaServer)
```



4. 카프카 예제

a. 카프카 토픽 생성

주키퍼, 카프카 실행한 상태에서 새로운 cmd 실행 ->

kafka 설치 경로로 이동(ex. cd C:\kafka_2.13-2.8.0) ->

bin\windows\kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test20220525 입력 ->

Created topic "test20220525" 가 나오면 정상 ->

```
명령 프롬프트
Microsoft Windows [Version 10.0.19044.1706]
(c) Microsoft Corporation. All rights reserved.

C:\Users\W82109>cd C:\kafka_2.13-2.8.0

C:\kafka_2.13-2.8.0>bin\windows\kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test20220525
Created topic test20220525.

C:\kafka_2.13-2.8.0>
```

b. 카프카 토픽 리스트 조회

bin\windows\kafka-topics.bat --list --zookeeper localhost:2181 입력 ->

test20220525 가 나오면 정상

```
C:\kafka_2.13-2.8.0>bin\windows\kafka-topics.bat --list --zookeeper localhost:2181
test20220525
```

c. 카프카 컨슈머 시작

bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test20220525 입력 -> 아무 변화가 없어야 정상

- config : bootstrap → 카프카 클러스터에 대한 초기 커넥션을 구축하는 데 사용할 호스트/포트 쌍 리스트

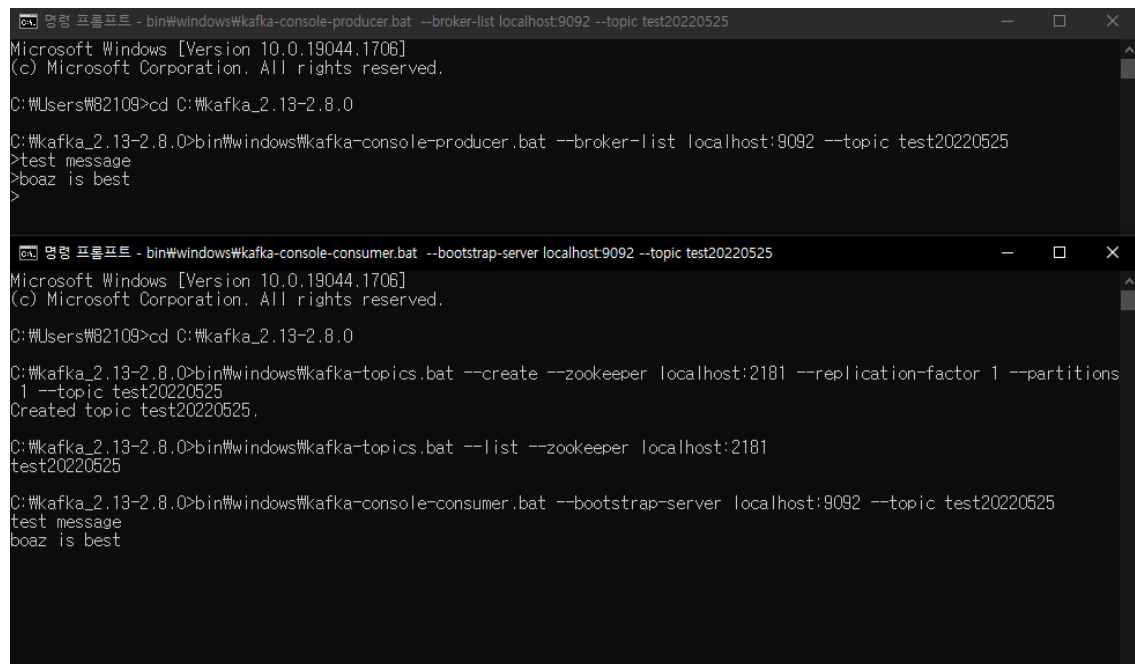
```
C:\kafka_2.13-2.8.0>bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test20220525
```

d. 카프카 프로듀서 시작

주키퍼, 카프카, 컨슈머 cmd창 유지한 채로 새로운 cmd 실행 ->

kafka 설치 경로로 이동 (ex. cd C:\kafka_2.13-2.8.0) ->

bin\windows\kafka-console-producer.bat --broker-list localhost:9092 --topic test20220525 입력-> 메시지 입력 후 consumer cmd 창에서 메시지 수신 확인

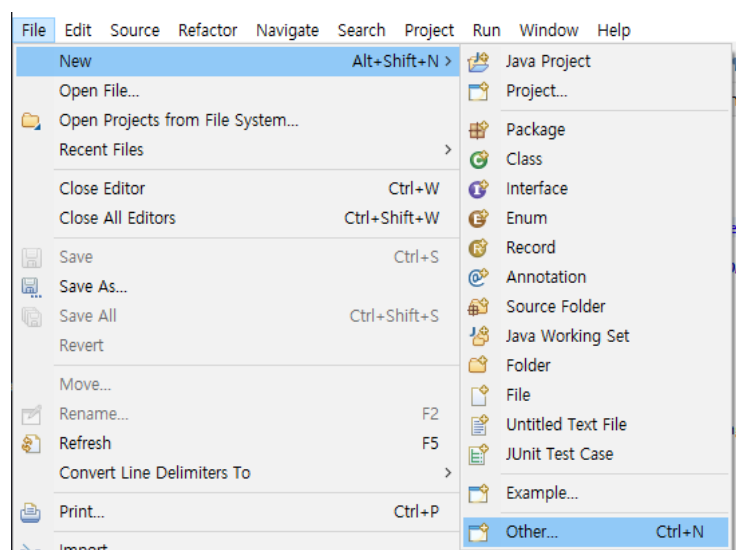


The first screenshot shows a command prompt window titled '명령 프롬프트 - bin\windows\kafka-console-producer.bat --broker-list localhost:9092 --topic test20220525'. The user navigates to 'C:\kafka_2.13-2.8.0' and runs 'kafka-console-producer.bat --broker-list localhost:9092 --topic test20220525'. They then enter 'test message' and 'boaz is best' on separate lines.

The second screenshot shows a command prompt window titled '명령 프롬프트 - bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test20220525'. The user navigates to 'C:\kafka_2.13-2.8.0' and runs 'kafka-topics.bat --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test20220525'. They then run 'kafka-topics.bat --list --zookeeper localhost:2181' and see 'test20220525'. Finally, they run 'kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic test20220525' and receive the messages 'test message' and 'boaz is best'.

5. 메이븐과 이클립스를 이용한 카프카 클러스터 구축

a. 프로젝트 생성



Select a wizard

Create a Maven project



Wizards:

type filter text

- > General
- > Git
- > Gradle
- > Java
- ▼ Maven
 - Check out Maven Projects from SCM
 - Maven Module
 - Maven Project
- > Oomph
- > XML
- > Examples



< Back

Next >

Finish

Cancel

New Maven project

Select project name and location



☒ Create a simple project (skip archetype selection)

☒ Use default Workspace location

Location:

C:\#



Browse...

☐ Add project(s) to working set

Working set:



More...

► Advanced



< Back

Next >

Finish

Cancel

New Maven project
Configure project

Artifact

Group Id:

Artifact Id:

Version:

Packaging:

Name:

Description:

Parent Project

Group Id:

Artifact Id:

Version:

▶ Advanced

b. dependency 추가

pom.xml에 의존성 추가 -> 저장



pom.xml

빌드 파일을 사용하여 빌드 정보 기술

→ project 내에 모든 정보를 기술

⇒ dependency 추가 : 이 프로그램이 참조하는 라이브러리 추가

```
<dependencies>
  <dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-clients</artifactId>
    <version>1.1.0</version>
  </dependency>
  <dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka-streams</artifactId>
    <version>1.1.0</version>
  </dependency>
  <dependency>
    <groupId>org.apache.kafka</groupId>
    <artifactId>kafka_2.11</artifactId>
    <version>0.8.2.1</version>
  </dependency>
</dependencies>
```

```

1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http:
2 <modelVersion>4.0.0</modelVersion>
3 <groupId>com.home.project</groupId>
4 <artifactId>test1</artifactId>
5 <version>0.0.1-SNAPSHOT</version>
6 <packaging>war</packaging>
7 <name>test1</name>
8 <description>test maven project and kafka</description>
9 <dependencies>
10 <dependency>
11 <groupId>org.apache.kafka</groupId>
12 <artifactId>kafka-clients</artifactId>
13 <version>1.1.0</version>
14 </dependency>
15 <dependency>
16 <groupId>org.apache.kafka</groupId>
17 <artifactId>kafka-streams</artifactId>
18 <version>1.1.0</version>
19 </dependency>
20 <dependency>
21 <groupId>org.apache.kafka</groupId>
22 <artifactId>kafka_2.11</artifactId>
23 <version>0.8.2.1</version>
24 </dependency>
25 </dependencies>
26 </project>

```

c. consumer 작성

bootstrap.servers	카프카 클러스터에 대한 초기 커넥션을 구축하는 데 사용할 호스트/포트 쌍리스트
session.timeout.ms	카프카의 그룹 관리 기능에서 클라이언트 실패를 감지할 때 사용할 타임아웃(하트비트)
group.id	이 컨슈머가 속한 컨슈머 그룹을 식별하는 유니크 문자열



key.deserializer/value.deserializer

바이트로 표현된 Key, Value 값을 다시 객체로 만들어 주는 클래스

```

package com.test.kafka.meta;
import org.apache.kafka.clients.consumer.ConsumerRecord;
import org.apache.kafka.clients.consumer.ConsumerRecords;
import org.apache.kafka.clients.consumer.KafkaConsumer;
import java.util.Arrays;
import java.util.Properties;

public class consumer {

    public static void main(String[] args) {
        Properties configs = new Properties();
        // 환경 변수 설정
        configs.put("bootstrap.servers", "localhost:9092"); // kafka server host 및 port
        configs.put("session.timeout.ms", "10000"); // session 설정
        configs.put("group.id", "test20220525"); // topic 설정
        configs.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeserializer"); // key deserializer
        configs.put("value.deserializer", "org.apache.kafka.common.serialization.StringDeserializer"); // value deserializer

        KafkaConsumer<String, String> consumer = new KafkaConsumer<String, String>(configs); // consumer 생성

        consumer.subscribe(Arrays.asList("test20220525")); // topic 설정

        while (true) { // 계속 loop를 돌면서 producer의 message를 띄운다.
            ConsumerRecords<String, String> records = consumer.poll(500);
            for (ConsumerRecord<String, String> record : records) {
                String s = record.topic();
                if ("test20220525".equals(s)) {
                    System.out.println(record.value());
                } else {
                    throw new IllegalStateException("get message on topic " + record.topic());
                }
            }
        }
    }
}

```

```

1 package com.test.kafka.meta;
2 import org.apache.kafka.clients.consumer.ConsumerRecord;
7
8 public class consumer {
9
10     public static void main(String[] args) {
11         Properties configs = new Properties();
12         // 환경 변수 설정
13         configs.put("bootstrap.servers", "localhost:9092"); // kafka server host 및 port
14         configs.put("session.timeout.ms", "10000"); // session 설정
15         configs.put("group.id", "test20190715"); // topic 설정
16         configs.put("key.deserializer", "org.apache.kafka.common.serialization.StringDeserializer"); // key deserializer
17         configs.put("value.deserializer", "org.apache.kafka.common.serialization.StringDeserializer"); // value deserializer
18
19         KafkaConsumer<String, String> consumer = new KafkaConsumer<String, String>(configs); // consumer 생성
20
21         consumer.subscribe(Arrays.asList("test20220525")); // topic 설정
22
23         while (true) { // 계속 loop를 돌면서 producer의 message를 띄운다.
24             ConsumerRecords<String, String> records = consumer.poll(500);
25             for (ConsumerRecord<String, String> record : records) {
26                 String s = record.topic();
27                 if ("test20220525".equals(s)) {
28                     System.out.println(record.value());
29                 } else {
30                     throw new IllegalStateException("get message on topic " + record.topic());
31                 }
32             }
33         }
34     }
35 }

```

d. producer 작성

acks	프로듀서에서 요청을 완료한 것으로 간주하기 위해 필요한 리더의 승인(acknowledgment) 수 acks=all → 리더는 in-sync 레플리카 셋 전체가 레코드를 승인할 때까지 기다림
block.on.buffer.full	메모리 버퍼가 소진되면 새로운 레코드를 받지 않고 블럭하거나 에러를 던진다. 기본 false



key.serializer/value.serializer

카프카 메시지와 데이터 값들을 바이트 배열로 만들어 줄 클래스

```

package com.test.kafka.meta;

import java.io.IOException;
import java.util.Properties;
import org.apache.kafka.clients.producer.KafkaProducer;
import org.apache.kafka.clients.producer.ProducerRecord;

public class Producer {
    public static void main(String[] args) throws IOException {

        Properties configs = new Properties();
        configs.put("bootstrap.servers", "localhost:9092"); // kafka host 및 server 설정
        configs.put("acks", "all"); // 자신이 보낸 메시지에 대해 카프카로부터 확인을 기다리지 않습니다.
        configs.put("block.on.buffer.full", "true"); // 서버로 보낸 레코드를 버퍼링 할 때 사용할 수 있는 전체 메모리의 바이트수
        configs.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer"); // serialize 설정
        configs.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer"); // serialize 설정

        // producer 생성
        KafkaProducer<String, String> producer = new KafkaProducer<String, String>(configs);
        // message 전달
        for (int i = 0; i < 5; i++) {
            String v = "hello"+i;
            producer.send(new ProducerRecord<String, String>("test20120525", v));
        }
        // 종료
        producer.flush();
        producer.close();
    }
}

```

```

1 package com.test.kafka.meta;
2
3 import java.io.IOException;
4
5
6
7
8 public class Producer {
9     public static void main(String[] args) throws IOException {
10
11         Properties configs = new Properties();
12         configs.put("bootstrap.servers", "localhost:9092"); // kafka host 및 server 설정
13         configs.put("acks", "all"); // 자신이 보낸 메시지에 대해 카프카로부터 확인을 기다리지 않습니다.
14         configs.put("block.on.buffer.full", "true"); // 서버로 보낼 레코드를 버퍼링 할 때 사용할 수 있는 전체 메모리의 바이트수
15         configs.put("key.serializer", "org.apache.kafka.common.serialization.StringSerializer"); // serialize 설정
16         configs.put("value.serializer", "org.apache.kafka.common.serialization.StringSerializer"); // serialize 설정
17
18         // producer 생성
19         KafkaProducer<String, String> producer = new KafkaProducer<String, String>(configs);
20         // message 전달
21         for (int i = 0; i < 5; i++) {
22             String v = "hello"+i;
23             producer.send(new ProducerRecord<String, String>("test20120525", v));
24         }
25         // 종료
26         producer.flush();
27         producer.close();
28     }
29 }

```

e. 실행

주키퍼 실행 → 카프카 실행 → Run Producer → Run Consumer

```

Problems Javadoc Declarati
consumer [Java Application]
log4j:WARN No appenders could b
log4j:WARN Please initialize th
hello0
hello1
hello2
hello3
hello4

```

- 로그 보고 싶다면 main 첫 줄에 `BasicConfigurator.configure();` 넣어서 실행

reference

windows 환경에서 zookeeper 설치 및 실행

<http://www.apache.org/dyn/closer.cgi/zookeeper/> 링크 클릭(컨트롤 클릭) -> 공식 다운로드 사이트 or 미러 사이트에서 zookeeper 압축 파일 다운로드 압축 해제 -> 경로확인 (ex) C:\dev\zookeeper-3.4.14 -> data 경로 생성 (ex) C:\dev\zookeeper-3.4.14\data -> C:\dev\zookeeper-3.4.14\conf 경로에 있는 zoo_sample.cfg 복사 붙이기
 :: <https://man-tae.tistory.com/3>

<http://apache.tt.co.kr/zookeeper/>

Other mirror sites are suggested below.

It is essential that you verify the integrity of the downloaded hash (.md5 or .sha* file).

Please only use the backup mirrors to download KEYS, PG, other mirrors are working.

windows 환경에서 kafka 설치 및 실행

1. 카프카 설치 <https://kafka.apache.org/downloads> 링크 클릭(컨트롤 클릭) -> 공식 다운로드 사이트 or 미러 사이트에서 카프카 압축 파일 다운로드 -> 압축 해제 -> 2. 카프카 설정 디렉토리 config로..

:: <https://man-tae.tistory.com/5>

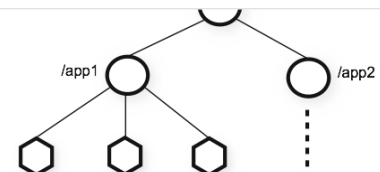
1.1.0

- Released March 28, 2018
- [Release Notes](#)
- Source download: [kafka-1.1.0-src.tgz \(asc, sha512\)](#)
- Binary downloads:
 - Scala 2.11 - [kafka_2.11-1.1.0.tgz \(asc, sha512\)](#)

apache Zookeeper 시작하기 (tutorial / windows 10)

분산시스템 환경에서 사용하기 적합한 key/value 저장 시스템입니다. 혹은 다른 더 적합한 대목을 달 수 있습니다. 사용하기 나쁩니다. znode 라는 데이터 저장 객체를 제공합니다. (key/value 와 흡사한)이 객체에 데이터를 넣고 빼는 기능을 제공합니다. 그림과 같이 디렉토리 형식을 사용하며, 데이터를 계층화한 구조로 저장합니다.어떤식으로 데


<https://lejewk.github.io/zookeeper-get-started/>



<https://go-coding.tistory.com/84>

[Maven] pom.xml 파일 기본


Maven은 커맨드를 사용하여 간단히 프로젝트를 만들거나 빌드가 가능하다. 예제로 만든 프로젝트는 단지 App.java 라는 소스 코드 파일이 있는 만큼 간단한 것이었다. Maven의 강점은 다양한 라이브러리와 프레임워크 등을 이용하는 경우도 그들을 모두 Maven이 관리해주는 점이다. 이러한 "프로젝트 관리"를 하기 위해서는, 단지 Maven 명령을 실행

 <https://araikuma.tistory.com/447>



Producer Configuration

아파치 카프카 공식 레퍼런스 를 한글로 번역한 문서입니다. 전체 목차는 여기 에 있습니다. 다음은 프로듀서의 설정 이다: key.serializer org.apache.kafka.common.serialization.Serializer 인터페이스를 구현한 키의 시리얼라이저 클래스 value.serializer org.apache.kafka.common.serialization.Serializer 인터페이스를 구현한 value의 시리얼

 <https://godekdls.github.io/Apache%20Kafka/producer-configuration/>



[Kafka] Maven project 로 Producer와 Consumer 작성해보기 (주석설명추가)

```
package KafkaProducer.producer; import org.apache.kafka.clients.consumer.ConsumerRecord; import
org.apache.kafka.clients.consumer.ConsumerRecords; import
org.apache.kafka.clients.consumer.KafkaConsumer; import java.util.Arrays; import java.util.Properties;
```

 <https://heeonii.tistory.com/22?category=935261>



카프카 프로듀서(Producer) 0.9.0 설정

프로듀서 설정은 config/producer.properties 파일에서.. bootstrap.servers 호스트/포트 쌍으로 된 리스트로 카프카 초기에 클러스터에 연결할 때 사용한다. 이 리스트는 단순히 처음에 전체 서버를 찾는데 사용되는 호스트 리스트다 - host1:port1,host2:port2.. 형식 전체 클러스터 멤버(클러스터 멤버는 동적으로 바뀐다)를 찾는 초기 커넥션으로 사용하기 때문에, 모든 서버 리스트를 포함할 필요는 없다.

 <https://free-strings.blogspot.com/2016/04/producer.html>