

Java Programming

Lab Kafka 2

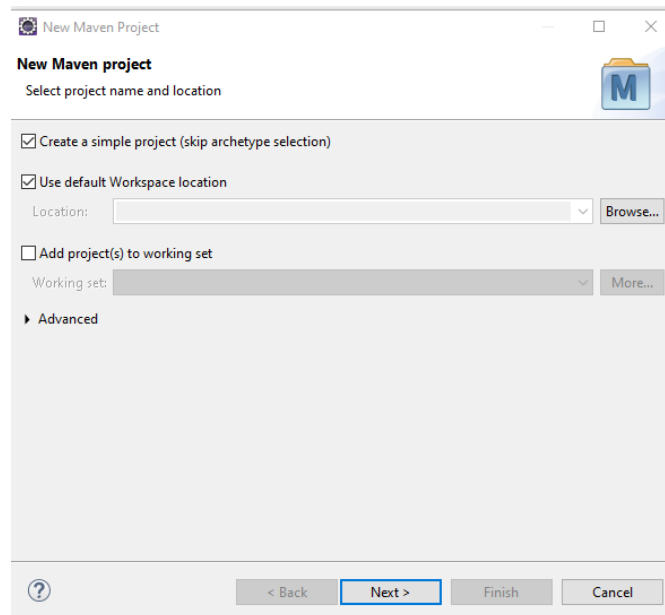
1. Introduction

You will create a Java Producer for a Kafka Topic.

2. Setup

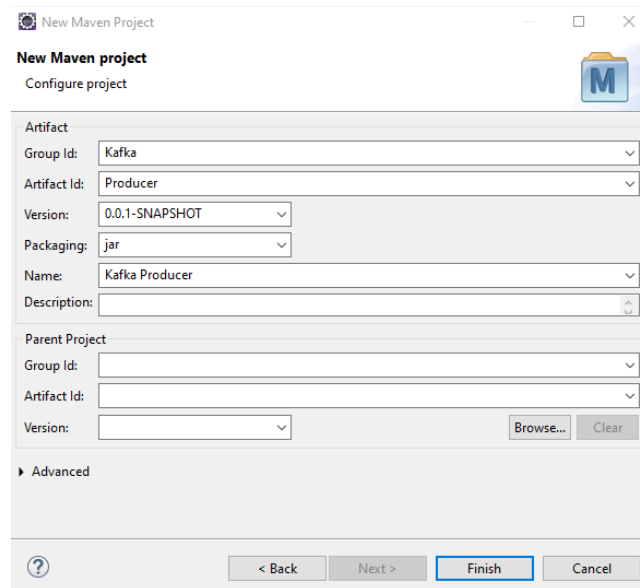
We need to use the Java Kafka API. The simplest way to do this is to create a Maven project and let Maven pull in the Java Kafka API jars.

To do this, start up a new workspace in eclipse and create a Maven project. Make sure the simple project is selected and the default location is selected.



Then select next to get to the next screen (next page)

Provide some Maven co-ordinates and select finish.



New Maven Project

Configure project

Artifact

Group Id: Kafka

Artifact Id: Producer

Version: 0.0.1-SNAPSHOT

Packaging: jar

Name: Kafka Producer

Description:

Parent Project

Group Id:

Artifact Id:

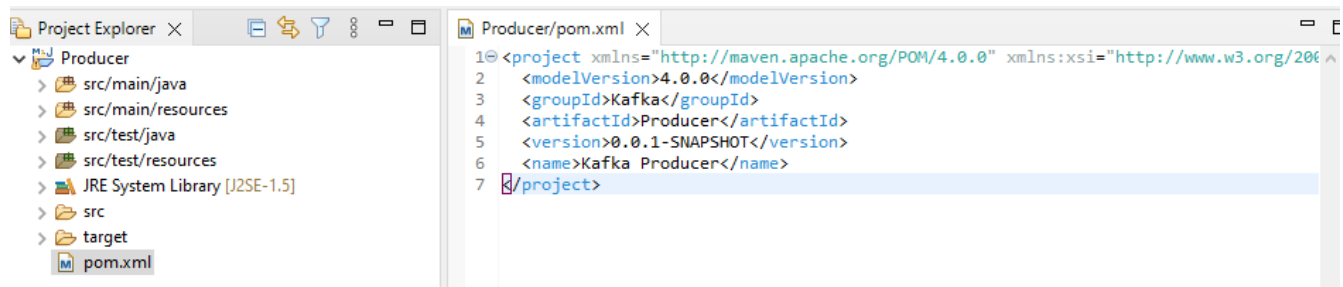
Version:

Browse... Clear

Advanced

< Back Next > Finish Cancel

In the resulting project directory is a POM.xml file.



```
1 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
2   <modelVersion>4.0.0</modelVersion>
3   <groupId>Kafka</groupId>
4   <artifactId>Producer</artifactId>
5   <version>0.0.1-SNAPSHOT</version>
6   <name>Kafka Producer</name>
7 </project>
```

We are going to edit this so that Maven will pull down all the Kafka Java libraries for us. In the lab assets folder, there some an xml file called POMFragment.xml.

Copy the contents of this file and place it before the closing `</project>` tag so your POM.xml looks like this the image on the next page. If you want to format it, right mouse click on the code in the window, select source from the menu and then format.

```

1=<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3
2  <modelVersion>4.0.0</modelVersion>
3  <groupId>Kafka</groupId>
4  <artifactId>Producer</artifactId>
5  <version>0.0.1-SNAPSHOT</version>
6=<dependencies>
7=<dependency>
8    <groupId>org.apache.kafka</groupId>
9    <artifactId>kafka-clients</artifactId>
10   <version>3.3.1</version>
11 </dependency>
12 <!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-api -->
13=<dependency>
14   <groupId>org.slf4j</groupId>
15   <artifactId>slf4j-api</artifactId>
16   <version>2.0.6</version>
17 </dependency>
18 <!-- https://mvnrepository.com/artifact/org.slf4j/slf4j-nop -->
19=<dependency>
20   <groupId>org.slf4j</groupId>
21   <artifactId>slf4j-nop</artifactId>
22   <version>2.0.6</version>
23
24 </dependency>
25 </dependencies>
26
27=<properties>
28   <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
29   <maven.compiler.source>18</maven.compiler.source>
30   <maven.compiler.target>18</maven.compiler.target>
31 </properties>
32 </project>

```

There are still a couple of issues that you might have to change. Line 29 specifies that the project is using Java 18, which is the version on the machine this lab was prepared on. If you are running a LearnQuest VM, then change the version from 18 to 17. If you are running on your own machine, change it to the version of Java you are running.

Once you have saved your POM.xml, under the project menu, select “Update Maven Project” for the changes to take effect.

You should see all of the jar files now in the Maven dependencies folder.

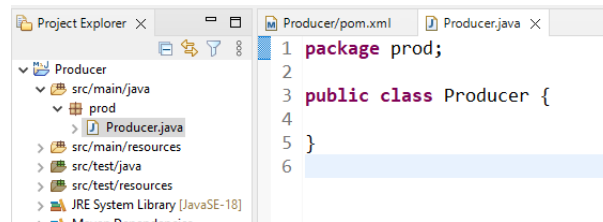
```

v Maven Dependencies
> kafka-clients-3.3.1.jar - C:\Users\micro\.m2\repository\org\apache\kafka\kafka-clients-3.3.1.jar
> zstd-jni-1.5.2-1.jar - C:\Users\micro\.m2\repository\com\github\luben\zstd-jni-1.5.2-1.jar
> lz4-java-1.8.0.jar - C:\Users\micro\.m2\repository\org\lz4\lz4-java\1.8.0
> snappy-java-1.1.8.4.jar - C:\Users\micro\.m2\repository\org\xerial\snappy\snappy-java-1.1.8.4.jar
> slf4j-api-2.0.6.jar - C:\Users\micro\.m2\repository\org\slf4j\slf4j-api\2.0.6
> slf4j-nop-2.0.6.jar - C:\Users\micro\.m2\repository\org\slf4j\slf4j-nop\2.0.6

```

3. Create the Producer Class

Create a prod package in the “src/main/java” folder and then create a Producer class in that package.



Since this is a simple producer, all of the code will be done in the main() method

4. Configure the Properties

Create a properties object and supply the Kafka configuration

```
public static void main(String[] args) {

    // Create properties object
    Properties prop = new Properties();
    prop.setProperty("bootstrap.servers", "localhost:9092");
    prop.setProperty("key.serializer", StringSerializer.class.getName());
    prop.setProperty("value.serializer", StringSerializer.class.getName());
```

5. Create the producer and producer record

We will send 10 messages of the form “value n” to the lab1 topic you created in lab one.

```
// Create the ProducerRecord
for (int i = 1 ; i <=10; i++) {
    ProducerRecord<String,String> record = new ProducerRecord<>("lab1","value "+i);

    // Send data
    p.send(record);
    System.out.println("Sent message " + i);
}
// flush and close the producer
p.flush();
p.close();
```

6. Set up Kafka

We need to have a Kafka broker running so go back to one and do the steps to:

1. Start Zookeeper in a command window
2. Start the Kafka Server in another window
3. And in a third window, check to see that you have the lab1 topic still in the broker. If not you can create it or another topic like you did in the previous lab

```
C:\kafka>.\bin\windows\kafka-topics.bat --list --bootstrap-server localhost:9092
__consumer_offsets
lab1
lab2
test
```

4. Now set up a console consumer like you did in the last class. As our producer pushes messages to the topic, we should be able to see them in the console producer.

```
C:\kafka>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic lab1
```

7. Run the producer

From the eclipse workspace, run the Producer code by running it as a Java Application.

Once it has finished, check the console consumer to see messages that have been sent.

```
C:\kafka>.\bin\windows\kafka-console-consumer.bat --bootstrap-server localhost:9092 --topic lab1
value 1
value 2
value 3
value 4
value 5
value 6
value 7
value 8
value 9
value 10
```

8. Shutdown

Unless you are going to immediately do Lab 2, you should probably shut down Kafka. In you are a VM, the cluster may crash or be corrupted in the VM goes into sleep mode.

End of Lab