Reference links

<https://avro.apache.org/docs/current/>

<https://github.com/joelittlejohn/jsonschema2pojo/wiki>

Convert JSON schema to POJO class

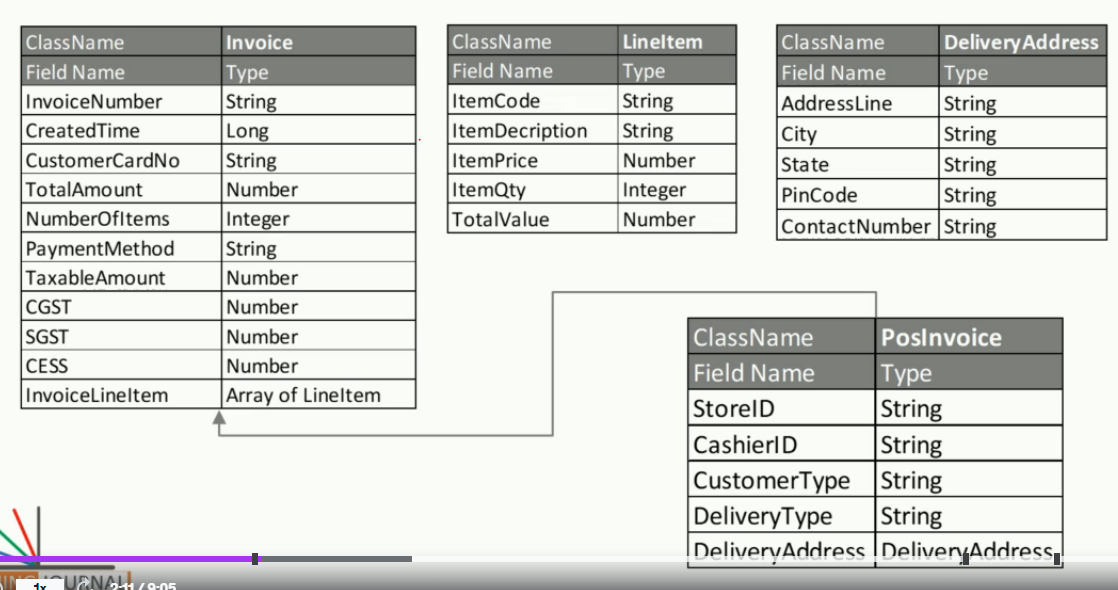
<https://github.com/joelittlejohn/jsonschema2pojo/wiki>

There is a jar called json schema to pojo use that jar and convert

<groupId>org.jsonschema2pojo</groupId>

<artifactId>jsonschema2pojo-maven-plugin</artifactId>

<version>1.1.3</version>



I still didn’t understand why to create a json object that to in below format

Instead directly we can create java classes instead of creating the json objects and use some bloody plugin generate java code

1. First prepare the Example json object

{

"type": "object",

"javaType": "guru.learningjournal.kafka.examples.types.Invoice",

"properties": {

"InvoiceNumber": {"type": "string"},

"CreatedTime": {"type": "object", "javaType": "java.lang.Long"},

"CustomerCardNo": {"type": "string"},

}

First we have to prepare this kind of json object, my suggestion instead of this better we can create java class directly after preparing the json we have to add the below plugin in pom.xml

1. Add 2 maven plugins- json plugin and maven compiler plugin in pom.xml and add related Jackson jars

<plugins>

<plugin>

<groupId>org.jsonschema2pojo</groupId>

<artifactId>jsonschema2pojo-maven-plugin</artifactId>

<version>1.1.3</version>

<plugin>  
 <groupId>org.apache.maven.plugins</groupId>  
 <artifactId>maven-compiler-plugin</artifactId>  
 <version>3.8.0</version>

Refer this page for full information<https://github.com/joelittlejohn/jsonschema2pojo/wiki/Getting-Started>

1. Then run the maven task called “compile” when u run this task then …. That will create the java classes
2. Refer project “09-json-to-pojo” project to and see full code

Creating custom serdes/ JsonSerde

Refer proj “11-pos-fanout-completed” to create a custom serdes class

By default IntegerSerde, FloatSerde are already present but Jsonserde is not present hence we need to create those serdes manually

If u are not satisfied with existing serdes then write all below custom classes

1. Class extending org.apache.kafka.common.serialization.Serializer (generally we don’t need as we already have Json,AvroSerializer classes with us)
2. Class extending org.apache.kafka.common.serialization.Deserializer (generally we don’t need as we already have Json,AvroDeSerializer classes with us)
3. class extending org.apache.kafka.common.serialization.Serde

ex:-



Create avsc file from pojo

In general its bit difficult to write json script file to create avsc file

implementation group: 'com.fasterxml.jackson.dataformat', name: 'jackson-dataformat-avro', version: '2.10.5'

Sample code to generate the avsc file from java class

//Currently below is not working  
ObjectMapper mapper=new ObjectMapper(new AvroFactory());  
AvroSchemaGenerator gen = new AvroSchemaGenerator();  
mapper.acceptJsonFormatVisitor(Employee.class, gen);  
AvroSchema schemaWrapper = gen.getGeneratedSchema();  
  
org.apache.avro.Schema avroSchema = schemaWrapper.getAvroSchema();  
String asJson = avroSchema.toString(true);  
System.*out*.println(asJson);

1. using these plugins u can generate java classes from these .avsc files and vice versa
2. if u want to send data using Avro, then in properties use Avro serializer as below

properties.put(ProducerConfig.VALUE\_SERIALIZER\_CLASS\_CONFIG, KafkaAvroSerializer.class);

properties.put(AbstractKafkaAvroSerDeConfig.SCHEMA\_REGISTRY\_URL\_CONFIG, AppConfigs.schemaRegistryServers);

Avro to pojo

<https://avro.apache.org/docs/current/>

As of now avro wont support extending classes

If u want avro, then let all properties be in single class,

1. Create avro files .asc same like json files where we will create pojos for the avro
2. Add maven compiler plugin, maven avro plugin (for that group id of that plugin will be avro
3. Add avro jars (org.apache.avro- avro-1.8.1)

# Avsc avro to pojo

1. We should create an avro file as below and we should use plugin to generate the pojo file

{

"namespace": "guru.learningjournal.kafka.examples.types", //This is the target package in which all the generated classes will be stored

"type": "record",

"name": "LineItem",

"fields": [

{"name": "ItemCode","type": ["null","string"]},

{"name": "ItemDescription","type": ["null","string"]},

{"name": "ItemPrice","type": ["null","double"]},

{"name": "ItemQty","type": ["null","int"]},

{"name": "TotalValue","type": ["null","double"]}

]

}

1. Plugin to generate the pojos from avro avsc file

<build>

<plugins>

<!-- Maven Compiler Plugin-->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.8.0</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

<!-- Maven Avro plugin for generating pojo-->

<plugin>

<groupId>org.apache.avro</groupId>

<artifactId>avro-maven-plugin</artifactId>

<version>1.8.1</version>

<executions>

<execution>

<phase>generate-sources</phase>

<goals>

<goal>schema</goal>

</goals>

<configuration>

<sourceDirectory>${project.basedir}/src/main/resources/schema/</sourceDirectory>

<outputDirectory>${project.basedir}/src/main/java/</outputDirectory>

<imports>

<import>${project.basedir}/src/main/resources/schema/LineItem.avsc</import>

<import>${project.basedir}/src/main/resources/schema/DeliveryAddress.avsc</import>

</imports>

</configuration>

</execution>

</executions>

</plugin>

</plugins>

</build>