Big Data Project

Wikimedia recent change streaming parsing

Khassangali Gapparov 2022 @MIU

Technology stack

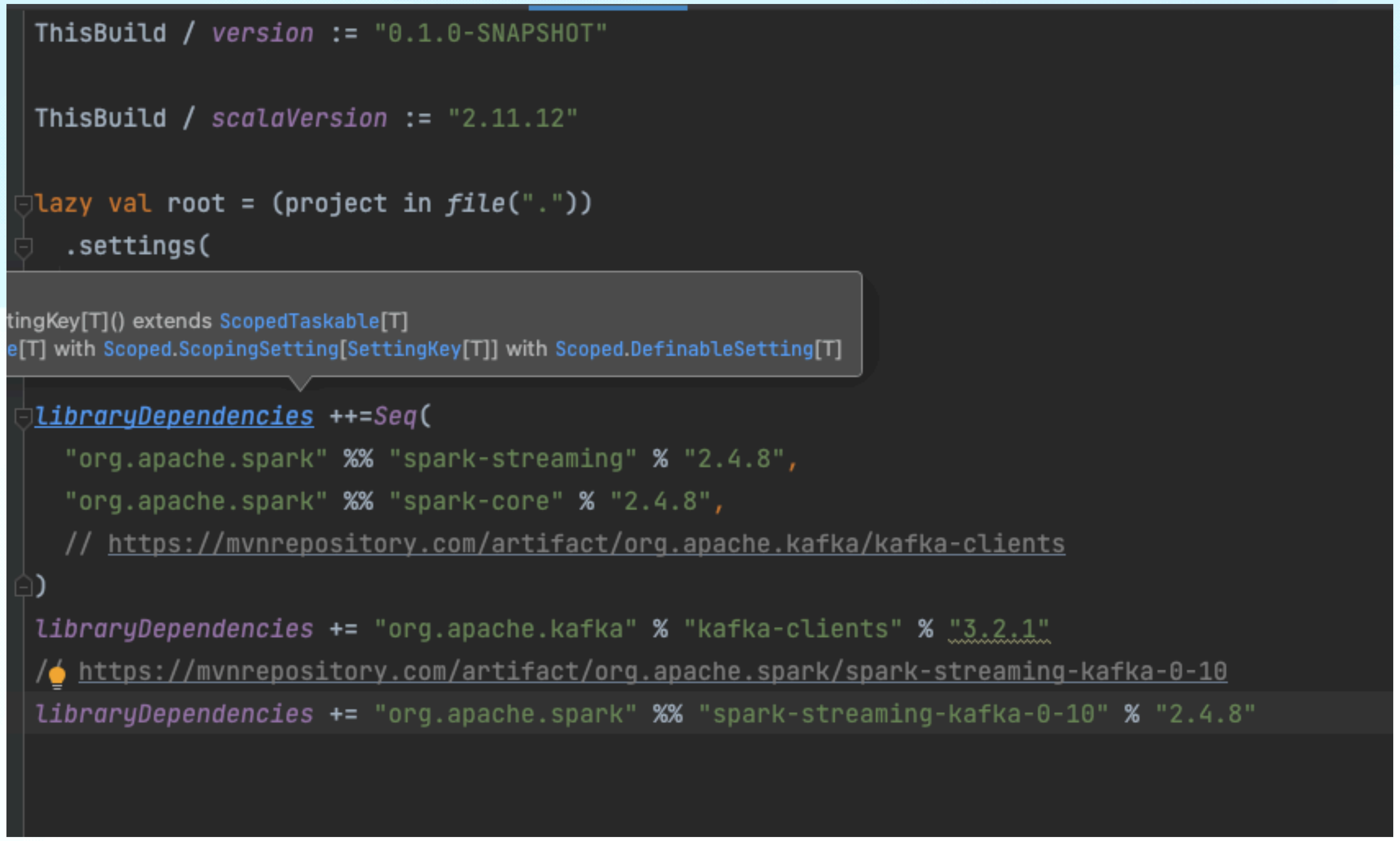
Overall main technologies list

- Streaming source from Wikimedia recent change: https://stream.wikimedia.org/v2/stream/recentchange
- Message queue tool - Kafka (local cluster)
- Streaming source event listener okHttp3.event source
- Spark with Streaming API
- HDFS cluster to save the final output
- IntelliJ Idea for creating java applications (part1 consumer, part2 producer)

Part1

Kafka Producer Application

- Created scala project with appropriate libraries:

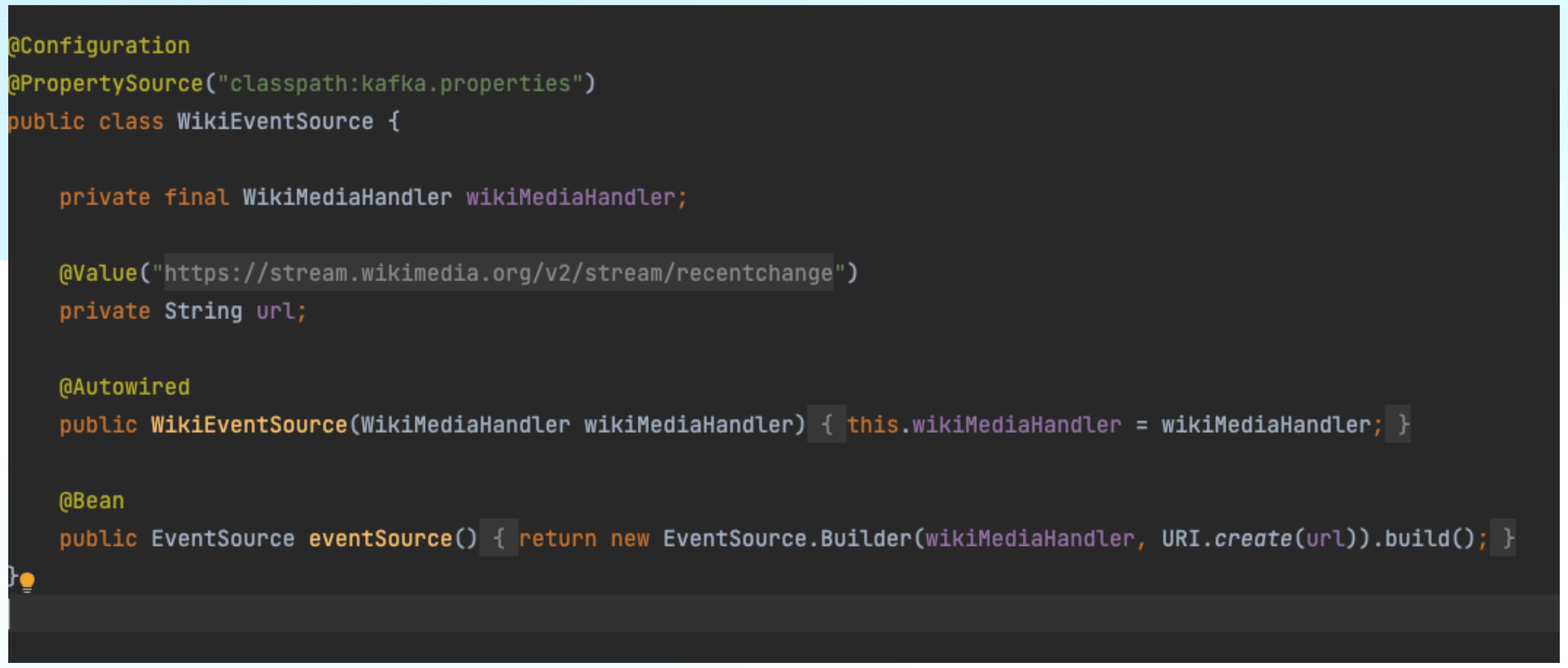


```
ThisBuild / version := "0.1.0-SNAPSHOT"

ThisBuild / scalaVersion := "2.11.12"

lazy val root = (project in file("."))
  .settings(

tingKey[T]() extends ScopedTaskable[T]
e[T] with Scoped.ScopingSetting[SettingKey[T]] with Scoped.DefinableSetting[T]

libraryDependencies ++=Seq(
  "org.apache.spark" %% "spark-streaming" % "2.4.8",
  "org.apache.spark" %% "spark-core" % "2.4.8",
  // https://mvnrepository.com/artifact/org.apache.kafka/kafka-clients
)
libraryDependencies += "org.apache.kafka" % "kafka-clients" % "3.2.1"
// https://mvnrepository.com/artifact/org.apache.spark/spark-streaming-kafka-0-10
libraryDependencies += "org.apache.spark" %% "spark-streaming-kafka-0-10" % "2.4.8"
```

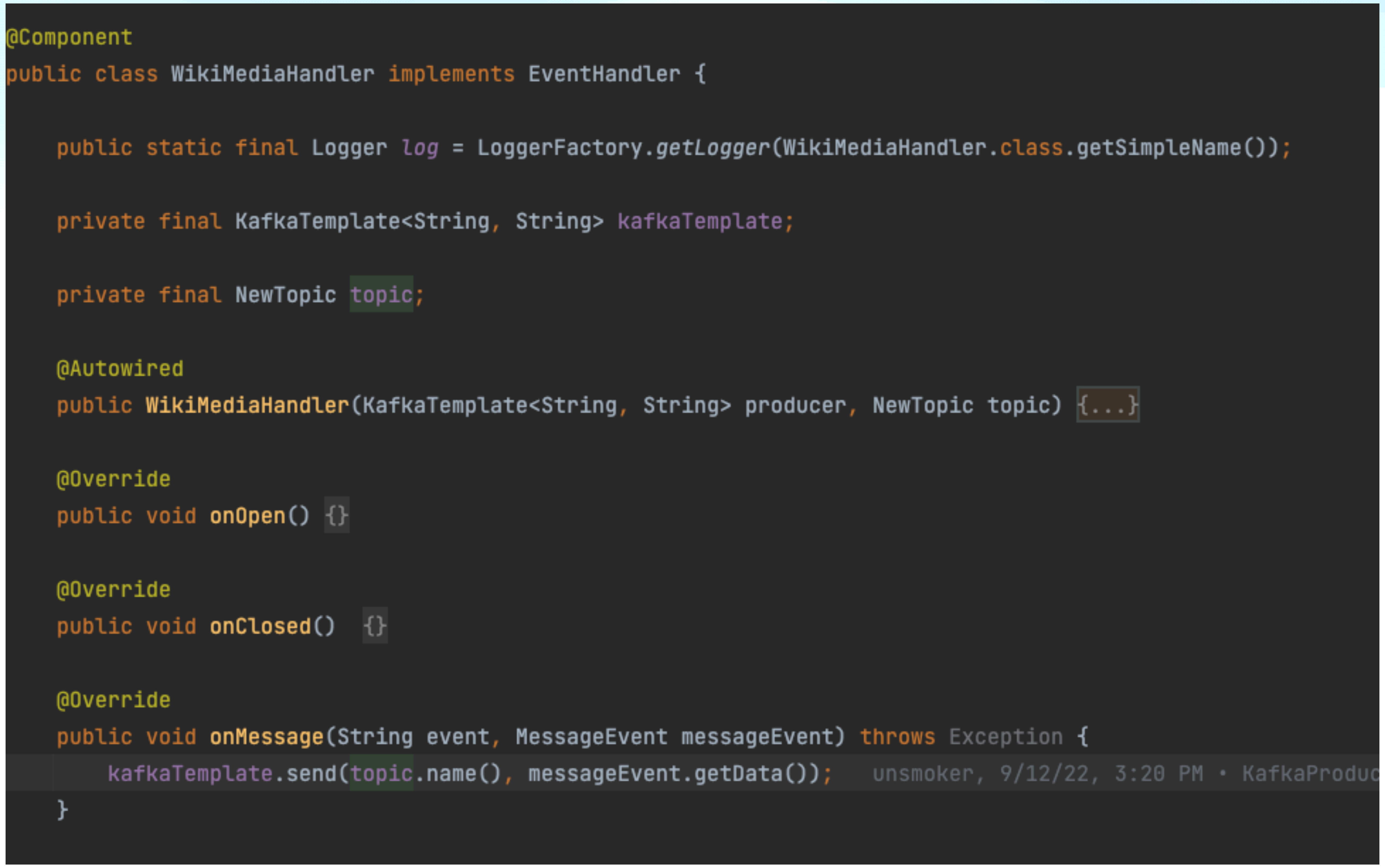
Part 1 cont…

Http event source

- Listen the URL for events and @onMessage we send message to Kafka producer with content of payload.



```java
@Configuration
@PropertySource("classpath:kafka.properties")
public class WikiEventSource {

    private final WikiMediaHandler wikiMediaHandler;

    @Value("https://stream.wikimedia.org/v2/stream/recentchange")
    private String url;

    @Autowired
    public WikiEventSource(WikiMediaHandler wikiMediaHandler) { this.wikiMediaHandler = wikiMediaHandler; }

    @Bean
    public EventSource eventSource() { return new EventSource.Builder(wikiMediaHandler, URI.create(url)).build(); }
}
```



```java
@Component
public class WikiMediaHandler implements EventHandler {

    public static final Logger log = LoggerFactory.getLogger(WikiMediaHandler.class.getSimpleName());

    private final KafkaTemplate<String, String> kafkaTemplate;

    private final NewTopic topic;

    @Autowired
    public WikiMediaHandler(KafkaTemplate<String, String> producer, NewTopic topic) {...}

    @Override
    public void onOpen() {}

    @Override
    public void onClosed()  {}

    @Override
    public void onMessage(String event, MessageEvent messageEvent) throws Exception {
        kafkaTemplate.send(topic.name(), messageEvent.getData());   unsmoker, 9/12/22, 3:20 PM • KafkaProduc
    }
```

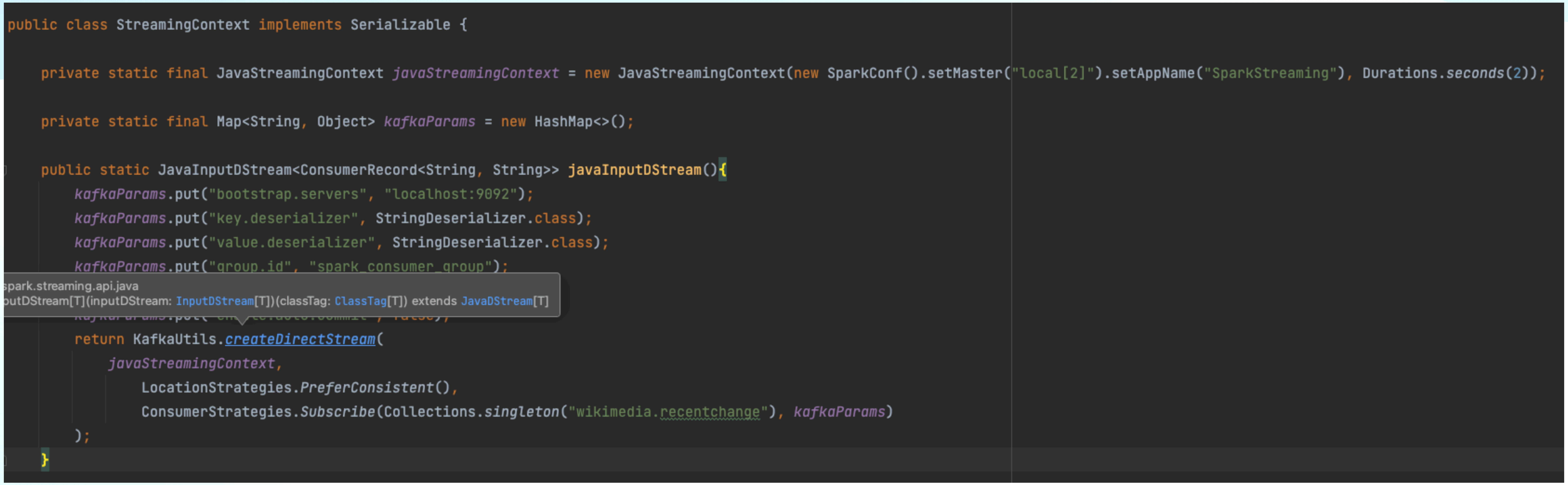
Part 2

Kafka Consumer

Part2

Kafka Consumer

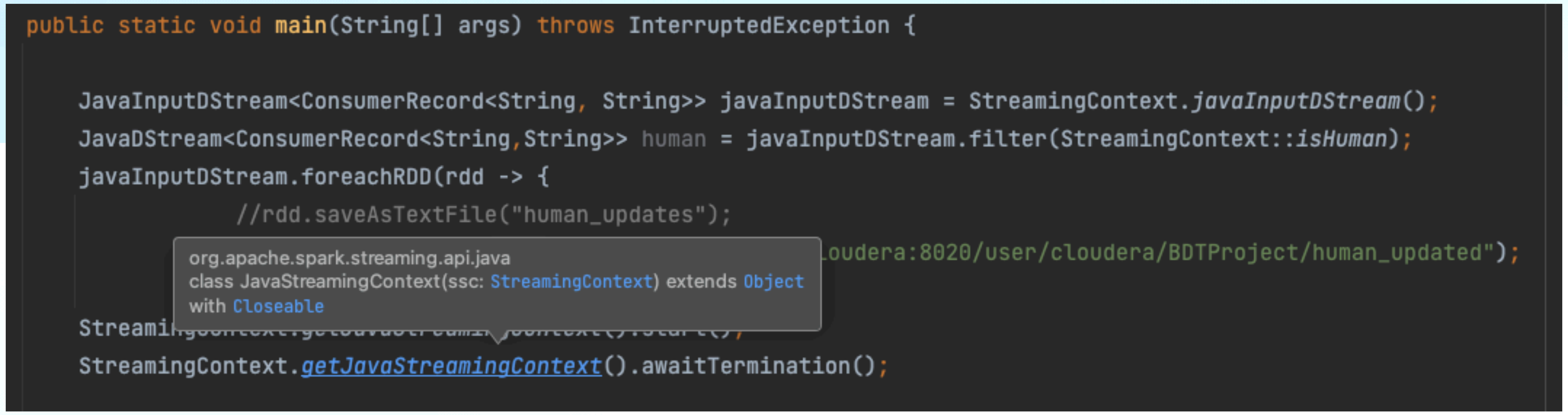
- We subscribe to the Kafka topic (“Wikimedia.recentchange”). with Kafka params shown as bellow configuration:



```
public class StreamingContext implements Serializable {

    private static final JavaStreamingContext javaStreamingContext = new JavaStreamingContext(new SparkConf().setMaster("local[2]").setAppName("SparkStreaming"), Durations.seconds(2));

    private static final Map<String, Object> kafkaParams = new HashMap<>();

    public static JavaInputDStream<ConsumerRecord<String, String>> javaInputDStream(){
        kafkaParams.put("bootstrap.servers", "localhost:9092");
        kafkaParams.put("key.deserializer", StringDeserializer.class);
        kafkaParams.put("value.deserializer", StringDeserializer.class);
        kafkaParams.put("group.id", "spark_consumer_group");
spark.streaming.api.java
putDStream[T](inputDStream: InputDStream[T])(classTag: ClassTag[T]) extends JavaDStream[T]
        [illegible]
        return KafkaUtils.createDirectStream(
            javaStreamingContext,
                LocationStrategies.PreferConsistent(),
                ConsumerStrategies.Subscribe(Collections.singleton("wikimedia.recentchange"), kafkaParams)
        );
    }
```

Kafka consumer

Spark RDD filtering and saving to HDFS

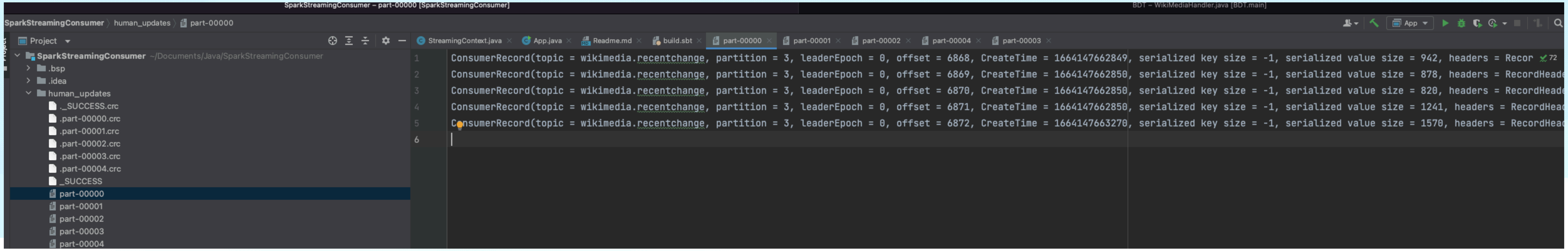
- Enable the stream and filter the output and save the final RDD result as text file.
- 

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

    JavaInputDStream<ConsumerRecord<String, String>> javaInputDStream = StreamingContext.javaInputDStream();
    JavaDStream<ConsumerRecord<String,String>> human = javaInputDStream.filter(StreamingContext::isHuman);
    javaInputDStream.foreachRDD(rdd -> {
                //rdd.saveAsTextFile("human_updates");
                                                                oudera:8020/user/cloudera/BDTProject/human_updated");

    Streami
    StreamingContext.getJavaStreamingContext().awaitTermination();
```

```
org.apache.spark.streaming.api.java
class JavaStreamingContext(ssc: StreamingContext) extends Object
with Closeable
```

Result

The final text records in file system.

- 

SparkStreamingConsumer – part-00000 [SparkStreamingConsumer]
BDT – WikiMediaHandler.java [BDT.main]
SparkStreamingConsumer 〉 human_updates 〉 part-00000
Project
StreamingContext.java
App.java
Readme.md
build.sbt
part-00000
part-00001
part-00002
part-00004
part-00003
App

```
SparkStreamingConsumer ~/Documents/Java/SparkStreamingConsumer
  .bsp
  .idea
  human_updates
    ._SUCCESS.crc
    .part-00000.crc
    .part-00001.crc
    .part-00002.crc
    .part-00003.crc
    .part-00004.crc
    _SUCCESS
    part-00000
    part-00001
    part-00002
    part-00003
    part-00004
```

```
ConsumerRecord(topic = wikimedia.recentchange, partition = 3, leaderEpoch = 0, offset = 6868, CreateTime = 1664147662849, serialized key size = -1, serialized value size = 942, headers = Recor
ConsumerRecord(topic = wikimedia.recentchange, partition = 3, leaderEpoch = 0, offset = 6869, CreateTime = 1664147662850, serialized key size = -1, serialized value size = 878, headers = RecordHead
ConsumerRecord(topic = wikimedia.recentchange, partition = 3, leaderEpoch = 0, offset = 6870, CreateTime = 1664147662850, serialized key size = -1, serialized value size = 820, headers = RecordHead
ConsumerRecord(topic = wikimedia.recentchange, partition = 3, leaderEpoch = 0, offset = 6871, CreateTime = 1664147662850, serialized key size = -1, serialized value size = 1241, headers = RecordHea
ConsumerRecord(topic = wikimedia.recentchange, partition = 3, leaderEpoch = 0, offset = 6872, CreateTime = 1664147663270, serialized key size = -1, serialized value size = 1570, headers = RecordHea
```