

```

package it.polito.bigdata.spark.Exercise47;

import org.apache.spark.streaming.Durations;
import org.apache.spark.streaming.api.java.JavaDStream;
import org.apache.spark.streaming.api.java.JavaPairDStream;
import org.apache.spark.streaming.api.java.JavaReceiverInputDStream;
import org.apache.spark.streaming.api.java.JavaStreamingContext;

import scala.Tuple2;

import org.apache.spark.SparkConf;
import org.apache.spark.api.java.JavaPairRDD;
import org.apache.spark.api.java.JavaSparkContext;

public class SparkDriver {

    public static void main(String[] args) {

        String inputFile;
        String outputPathPrefix;

        inputFile=args[0];
        outputPathPrefix=args[1];

        // Create a configuration object and set the name of the application
        SparkConf conf=
        new SparkConf().setAppName("Spark Streaming ....");

        // Create a Spark Streaming Context object
        JavaStreamingContext jssc =
        new JavaStreamingContext(conf, Durations.seconds(2));

        JavaSparkContext sc = jssc.sparkContext();

        // Create a (Receiver) DStream that will connect to localhost:9999
        JavaReceiverInputDStream<String> readings =
            jssc.socketTextStream("localhost", 9999);

        // Input stream
        // Each readings has the format:
        // stationId,#free slots,#used slots,timestamp
        // Select readings with num. free slots = 0
        JavaDStream<String> fullReadings =
            readings.filter(new Full());

        // <StationId, timestamp>
        JavaPairDStream<String,String> stationIdTime =
            fullReadings.mapToPair(new StationTimestamp());

        JavaPairRDD<String, String> stationName =
            sc.textFile(inputFile).mapToPair(new StationName());

        // <location, timestamp>
        final JavaDStream<Tuple2<String,String>> stationNameTime =
            stationIdTime.transform(new RetrieveName(stationName));
    }
}

```

```
stationNameTime.print();  
stationNameTime.dstream().saveAsTextFiles(outputPathPrefix, "");  
  
// Start the computation  
jssc.start();  
  
jssc.awaitTerminationOrTimeout(120000);  
  
jssc.close();  
  
    }  
}
```