package com.twitter.search.earlybird.factory;

import org.apache.kafka.clients.consumer.KafkaConsumer;

import com.twitter.search.common.indexing.thriftjava.ThriftVersionedEvents;

import com.twitter.search.common.util.io.kafka.CompactThriftDeserializer;

import com.twitter.search.common.util.io.kafka.FinagleKafkaClientUtils;

/\*\*

\* Responsible for creating kafka consumers.

\*/

public class ProductionEarlybirdKafkaConsumersFactory implements EarlybirdKafkaConsumersFactory {

private final String kafkaPath;

private final int defaultMaxPollRecords;

ProductionEarlybirdKafkaConsumersFactory(String kafkaPath, int defaultMaxPollRecords) {

this.kafkaPath = kafkaPath;

this.defaultMaxPollRecords = defaultMaxPollRecords;

}

/\*\*

\* Create a kafka consumer with set maximum of records to be polled.

\*/

@Override

public KafkaConsumer<Long, ThriftVersionedEvents> createKafkaConsumer(

String clientID, int maxPollRecords) {

return FinagleKafkaClientUtils.newKafkaConsumerForAssigning(

kafkaPath,

new CompactThriftDeserializer<>(ThriftVersionedEvents.class),

clientID,

maxPollRecords);

}

/\*\*

\* Create a kafka consumer with default records to be polled.

\*/

@Override

public KafkaConsumer<Long, ThriftVersionedEvents> createKafkaConsumer(String clientID) {

return createKafkaConsumer(clientID, defaultMaxPollRecords);

}

}