package com.twitter.unified\_user\_actions.client.summingbird

import com.twitter.summingbird.TimeExtractor

import com.twitter.summingbird.storm.Storm

import com.twitter.summingbird\_internal.sources.AppId

import com.twitter.summingbird\_internal.sources.SourceFactory

import com.twitter.tormenta\_internal.spout.Kafka2ScroogeSpoutWrapper

import com.twitter.unified\_user\_actions.client.config.ClientConfig

import com.twitter.unified\_user\_actions.thriftscala.UnifiedUserAction

import com.twitter.unified\_user\_actions.client.config.KafkaConfigs

case class UnifiedUserActionsSourceScrooge(

appId: AppId,

parallelism: Int,

kafkaConfig: ClientConfig = KafkaConfigs.ProdUnifiedUserActions,

skipToLatest: Boolean = false,

enableTls: Boolean = true)

extends SourceFactory[Storm, UnifiedUserAction] {

override def name: String = "UnifiedUserActionsSource"

override def description: String = "Unified User Actions (UUA) events"

// The event timestamps from summingbird's perspective (client), is our internally

// outputted timestamps (producer). This ensures time-continuity between the client and the

// producer.

val timeExtractor: TimeExtractor[UnifiedUserAction] = TimeExtractor { e =>

e.eventMetadata.receivedTimestampMs

}

override def source = {

Storm.source(

Kafka2ScroogeSpoutWrapper(

codec = UnifiedUserAction,

cluster = kafkaConfig.cluster.name,

topic = kafkaConfig.topic,

appId = appId.get,

skipToLatest = skipToLatest,

enableTls = enableTls

),

Some(parallelism)

)(timeExtractor)

}

}