package com.twitter.unified\_user\_actions.adapter.social\_graph\_event

import com.twitter.finagle.stats.NullStatsReceiver

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.finatra.kafka.serde.UnKeyed

import com.twitter.socialgraph.thriftscala.Action.\_

import com.twitter.socialgraph.thriftscala.WriteEvent

import com.twitter.socialgraph.thriftscala.{Action => SocialGraphAction}

import com.twitter.unified\_user\_actions.adapter.AbstractAdapter

import com.twitter.unified\_user\_actions.adapter.social\_graph\_event.SocialGraphEngagement.\_

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

class SocialGraphAdapter extends AbstractAdapter[WriteEvent, UnKeyed, UnifiedUserAction] {

import SocialGraphAdapter.\_

override def adaptOneToKeyedMany(

input: WriteEvent,

statsReceiver: StatsReceiver = NullStatsReceiver

): Seq[(UnKeyed, UnifiedUserAction)] =

adaptEvent(input).map { e => (UnKeyed, e) }

}

object SocialGraphAdapter {

def adaptEvent(writeEvent: WriteEvent): Seq[UnifiedUserAction] =

Option(writeEvent).flatMap { e =>

socialGraphWriteEventTypeToUuaEngagementType.get(e.action)

} match {

case Some(uuaAction) => uuaAction.toUnifiedUserAction(writeEvent, uuaAction)

case None => Nil

}

private val socialGraphWriteEventTypeToUuaEngagementType: Map[

SocialGraphAction,

BaseSocialGraphWriteEvent[\_]

] =

Map[SocialGraphAction, BaseSocialGraphWriteEvent[\_]](

Follow -> ProfileFollow,

Unfollow -> ProfileUnfollow,

Block -> ProfileBlock,

Unblock -> ProfileUnblock,

Mute -> ProfileMute,

Unmute -> ProfileUnmute,

ReportAsSpam -> ProfileReportAsSpam,

ReportAsAbuse -> ProfileReportAsAbuse

)

}