package com.twitter.unified\_user\_actions.adapter.user\_modification

import com.twitter.finagle.stats.NullStatsReceiver

import com.twitter.finagle.stats.StatsReceiver

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

import com.twitter.gizmoduck.thriftscala.UserModification

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

import com.twitter.unified\_user\_actions.adapter.user\_modification\_event.UserCreate

import com.twitter.unified\_user\_actions.adapter.user\_modification\_event.UserUpdate

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

class UserModificationAdapter

extends AbstractAdapter[UserModification, UnKeyed, UnifiedUserAction] {

import UserModificationAdapter.\_

override def adaptOneToKeyedMany(

input: UserModification,

statsReceiver: StatsReceiver = NullStatsReceiver

): Seq[(UnKeyed, UnifiedUserAction)] =

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

}

object UserModificationAdapter {

def adaptEvent(input: UserModification): Seq[UnifiedUserAction] =

Option(input).toSeq.flatMap { e =>

if (e.create.isDefined) { // User create

Some(UserCreate.getUUA(input))

} else if (e.update.isDefined) { // User updates

Some(UserUpdate.getUUA(input))

} else if (e.destroy.isDefined) {

None

} else if (e.erase.isDefined) {

None

} else {

throw new IllegalArgumentException(

"None of the possible events is defined, there must be something with the source")

}

}

}