package com.twitter.unified\_user\_actions.adapter.client\_event

import com.twitter.clientapp.thriftscala.ItemType

import com.twitter.clientapp.thriftscala.LogEvent

import com.twitter.clientapp.thriftscala.{Item => LogEventItem}

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

abstract class BaseNotificationTabClientEvent(actionType: ActionType)

extends BaseClientEvent(actionType = actionType) {

// itemType is `None` for Notification Tab events

override def isItemTypeValid(itemTypeOpt: Option[ItemType]): Boolean =

ItemTypeFilterPredicates.ignoreItemType(itemTypeOpt)

override def getUuaItem(

ceItem: LogEventItem,

logEvent: LogEvent

): Option[Item] = for {

notificationTabDetails <- ceItem.notificationTabDetails

clientEventMetadata <- notificationTabDetails.clientEventMetadata

notificationId <- NotificationClientEventUtils.getNotificationIdForNotificationTab(ceItem)

} yield {

clientEventMetadata.tweetIds match {

// if `tweetIds` contain more than one Tweet id, create `MultiTweetNotification`

case Some(tweetIds) if tweetIds.size > 1 =>

Item.NotificationInfo(

NotificationInfo(

actionNotificationId = notificationId,

content = NotificationContent.MultiTweetNotification(

MultiTweetNotification(tweetIds = tweetIds))

))

// if `tweetIds` contain exactly one Tweet id, create `TweetNotification`

case Some(tweetIds) if tweetIds.size == 1 =>

Item.NotificationInfo(

NotificationInfo(

actionNotificationId = notificationId,

content =

NotificationContent.TweetNotification(TweetNotification(tweetId = tweetIds.head))))

// if `tweetIds` are missing, create `UnknownNotification`

case \_ =>

Item.NotificationInfo(

NotificationInfo(

actionNotificationId = notificationId,

content = NotificationContent.UnknownNotification(UnknownNotification())

))

}

}

}