package com.twitter.home\_mixer.marshaller.timeline\_logging

import com.twitter.home\_mixer.model.HomeFeatures.AuthorIdFeature

import com.twitter.home\_mixer.model.HomeFeatures.SourceTweetIdFeature

import com.twitter.home\_mixer.model.HomeFeatures.SourceUserIdFeature

import com.twitter.home\_mixer.model.HomeFeatures.SuggestTypeFeature

import com.twitter.product\_mixer.component\_library.model.presentation.urt.UrtItemPresentation

import com.twitter.product\_mixer.component\_library.model.presentation.urt.UrtModulePresentation

import com.twitter.product\_mixer.core.functional\_component.marshaller.response.urt.metadata.GeneralContextTypeMarshaller

import com.twitter.product\_mixer.core.model.common.presentation.CandidateWithDetails

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.tweet.TweetItem

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.ConversationGeneralContextType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.GeneralContext

import com.twitter.product\_mixer.core.model.marshalling.response.urt.metadata.TopicContext

import com.twitter.timelines.service.{thriftscala => tst}

import com.twitter.timelines.timeline\_logging.{thriftscala => thriftlog}

object TweetDetailsMarshaller {

private val generalContextTypeMarshaller = new GeneralContextTypeMarshaller()

def apply(entry: TweetItem, candidate: CandidateWithDetails): thriftlog.TweetDetails = {

val socialContext = candidate.presentation.flatMap {

case \_ @UrtItemPresentation(timelineItem: TweetItem, \_) => timelineItem.socialContext

case \_ @UrtModulePresentation(timelineModule) =>

timelineModule.items.head.item match {

case timelineItem: TweetItem => timelineItem.socialContext

case \_ => Some(ConversationGeneralContextType)

}

}

val socialContextType = socialContext match {

case Some(GeneralContext(contextType, \_, \_, \_, \_)) =>

Some(generalContextTypeMarshaller(contextType).value.toShort)

case Some(TopicContext(\_, \_)) => Some(tst.ContextType.Topic.value.toShort)

case \_ => None

}

thriftlog.TweetDetails(

sourceTweetId = candidate.features.getOrElse(SourceTweetIdFeature, None),

socialContextType = socialContextType,

suggestType = candidate.features.getOrElse(SuggestTypeFeature, None).map(\_.name),

authorId = candidate.features.getOrElse(AuthorIdFeature, None),

sourceAuthorId = candidate.features.getOrElse(SourceUserIdFeature, None)

)

}

}