package com.twitter.home\_mixer.marshaller.request

import com.twitter.home\_mixer.model.request.FollowingProductContext

import com.twitter.home\_mixer.model.request.ForYouProductContext

import com.twitter.home\_mixer.model.request.ListRecommendedUsersProductContext

import com.twitter.home\_mixer.model.request.ListTweetsProductContext

import com.twitter.home\_mixer.model.request.ScoredTweetsProductContext

import com.twitter.home\_mixer.model.request.SubscribedProductContext

import com.twitter.home\_mixer.{thriftscala => t}

import com.twitter.product\_mixer.core.model.marshalling.request.ProductContext

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class HomeMixerProductContextUnmarshaller @Inject() (

deviceContextUnmarshaller: DeviceContextUnmarshaller) {

def apply(productContext: t.ProductContext): ProductContext = productContext match {

case t.ProductContext.Following(p) =>

FollowingProductContext(

deviceContext = p.deviceContext.map(deviceContextUnmarshaller(\_)),

seenTweetIds = p.seenTweetIds,

dspClientContext = p.dspClientContext

)

case t.ProductContext.ForYou(p) =>

ForYouProductContext(

deviceContext = p.deviceContext.map(deviceContextUnmarshaller(\_)),

seenTweetIds = p.seenTweetIds,

dspClientContext = p.dspClientContext,

pushToHomeTweetId = p.pushToHomeTweetId

)

case t.ProductContext.ListManagement(p) =>

throw new UnsupportedOperationException(s"This product is no longer used")

case t.ProductContext.ScoredTweets(p) =>

ScoredTweetsProductContext(

deviceContext = p.deviceContext.map(deviceContextUnmarshaller(\_)),

seenTweetIds = p.seenTweetIds,

servedTweetIds = p.servedTweetIds,

backfillTweetIds = p.backfillTweetIds

)

case t.ProductContext.ListTweets(p) =>

ListTweetsProductContext(

listId = p.listId,

deviceContext = p.deviceContext.map(deviceContextUnmarshaller(\_)),

dspClientContext = p.dspClientContext

)

case t.ProductContext.ListRecommendedUsers(p) =>

ListRecommendedUsersProductContext(

listId = p.listId,

selectedUserIds = p.selectedUserIds,

excludedUserIds = p.excludedUserIds,

listName = p.listName

)

case t.ProductContext.Subscribed(p) =>

SubscribedProductContext(

deviceContext = p.deviceContext.map(deviceContextUnmarshaller(\_)),

seenTweetIds = p.seenTweetIds,

)

case t.ProductContext.UnknownUnionField(field) =>

throw new UnsupportedOperationException(s"Unknown display context: ${field.field.name}")

}

}