package com.twitter.home\_mixer.product.list\_tweets.model

import com.twitter.adserver.thriftscala.HomeTimelineType

import com.twitter.adserver.thriftscala.TimelineRequestParams

import com.twitter.dspbidder.commons.{thriftscala => dsp}

import com.twitter.home\_mixer.model.HomeAdsQuery

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

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

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

import com.twitter.product\_mixer.component\_library.model.cursor.UrtOrderedCursor

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

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

import com.twitter.product\_mixer.core.pipeline.HasPipelineCursor

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.timelines.configapi.Params

case class ListTweetsQuery(

override val params: Params,

override val clientContext: ClientContext,

override val pipelineCursor: Option[UrtOrderedCursor],

override val requestedMaxResults: Option[Int],

override val debugOptions: Option[DebugOptions],

override val features: Option[FeatureMap],

override val listId: Long,

override val deviceContext: Option[DeviceContext],

override val dspClientContext: Option[dsp.DspClientContext])

extends PipelineQuery

with HasPipelineCursor[UrtOrderedCursor]

with HasListId

with HomeAdsQuery {

override val product: Product = ListTweetsProduct

override def withFeatureMap(features: FeatureMap): ListTweetsQuery =

copy(features = Some(features))

override val timelineRequestParams: Option[TimelineRequestParams] =

Some(TimelineRequestParams(homeTimelineType = Some(HomeTimelineType.HomeLatest)))

}