package com.twitter.home\_mixer.model

import com.twitter.adserver.thriftscala.RequestTriggerType

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

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

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

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

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

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

import com.twitter.product\_mixer.component\_library.model.query.ads.AdsQuery

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

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

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

/\*\*

\* These are for feeds needed for ads only.

\*/

trait HomeAdsQuery

extends AdsQuery

with PipelineQuery

with HasDeviceContext

with HasPipelineCursor[UrtOrderedCursor] {

private val featureToRequestTriggerType = Seq(

(GetInitialFeature, RequestTriggerType.Initial),

(GetNewerFeature, RequestTriggerType.Scroll),

(GetOlderFeature, RequestTriggerType.Scroll),

(PollingFeature, RequestTriggerType.AutoRefresh)

)

override val autoplayEnabled: Option[Boolean] = deviceContext.flatMap(\_.autoplayEnabled)

override def requestTriggerType: Option[RequestTriggerType] = {

val features = this.features.getOrElse(FeatureMap.empty)

featureToRequestTriggerType.collectFirst {

case (feature, requestType) if features.get(feature) => Some(requestType)

}.flatten

}

override val disableNsfwAvoidance: Option[Boolean] = Some(true)

}