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

import com.twitter.clientapp.{thriftscala => ca}

import com.twitter.goldfinch.api.AdsInjectionSurfaceAreas

import com.twitter.home\_mixer.candidate\_pipeline.ConversationServiceCandidatePipelineConfigBuilder

import com.twitter.home\_mixer.functional\_component.feature\_hydrator.RequestQueryFeatureHydrator

import com.twitter.home\_mixer.functional\_component.side\_effect.HomeScribeClientEventSideEffect

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

import com.twitter.home\_mixer.param.HomeMixerFlagName.ScribeClientEventsFlag

import com.twitter.home\_mixer.product.list\_tweets.decorator.ListConversationServiceCandidateDecorator

import com.twitter.home\_mixer.product.list\_tweets.model.ListTweetsQuery

import com.twitter.home\_mixer.util.CandidatesUtil

import com.twitter.inject.annotations.Flag

import com.twitter.logpipeline.client.common.EventPublisher

import com.twitter.product\_mixer.component\_library.feature\_hydrator.query.social\_graph.SGSFollowedUsersQueryFeatureHydrator

import com.twitter.product\_mixer.component\_library.gate.NonEmptyCandidatesGate

import com.twitter.product\_mixer.component\_library.premarshaller.urt.UrtDomainMarshaller

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.AddEntriesWithReplaceAndShowAlertInstructionBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.OrderedBottomCursorBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.OrderedGapCursorBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.OrderedTopCursorBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.ReplaceAllEntries

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.ReplaceEntryInstructionBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.ShowAlertInstructionBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.StaticTimelineScribeConfigBuilder

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.UrtMetadataBuilder

import com.twitter.product\_mixer.component\_library.selector.InsertAppendResults

import com.twitter.product\_mixer.component\_library.selector.UpdateSortCandidates

import com.twitter.product\_mixer.component\_library.selector.ads.AdsInjector

import com.twitter.product\_mixer.component\_library.selector.ads.InsertAdResults

import com.twitter.product\_mixer.core.functional\_component.common.SpecificPipelines

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.QueryFeatureHydrator

import com.twitter.product\_mixer.core.functional\_component.marshaller.TransportMarshaller

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

import com.twitter.product\_mixer.core.functional\_component.premarshaller.DomainMarshaller

import com.twitter.product\_mixer.core.functional\_component.selector.Selector

import com.twitter.product\_mixer.core.functional\_component.side\_effect.PipelineResultSideEffect

import com.twitter.product\_mixer.core.model.common.UniversalNoun

import com.twitter.product\_mixer.core.model.common.identifier.CandidatePipelineIdentifier

import com.twitter.product\_mixer.core.model.common.identifier.MixerPipelineIdentifier

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

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

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

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

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

import com.twitter.product\_mixer.core.pipeline.candidate.CandidatePipelineConfig

import com.twitter.product\_mixer.core.pipeline.candidate.DependentCandidatePipelineConfig

import com.twitter.product\_mixer.core.pipeline.mixer.MixerPipelineConfig

import com.twitter.timelines.render.{thriftscala => urt}

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class ListTweetsMixerPipelineConfig @Inject() (

listTweetsTimelineServiceCandidatePipelineConfig: ListTweetsTimelineServiceCandidatePipelineConfig,

conversationServiceCandidatePipelineConfigBuilder: ConversationServiceCandidatePipelineConfigBuilder[

ListTweetsQuery

],

listTweetsAdsCandidatePipelineBuilder: ListTweetsAdsCandidatePipelineBuilder,

requestQueryFeatureHydrator: RequestQueryFeatureHydrator[ListTweetsQuery],

sgsFollowedUsersQueryFeatureHydrator: SGSFollowedUsersQueryFeatureHydrator,

adsInjector: AdsInjector,

clientEventsScribeEventPublisher: EventPublisher[ca.LogEvent],

urtTransportMarshaller: UrtTransportMarshaller,

@Flag(ScribeClientEventsFlag) enableScribeClientEvents: Boolean)

extends MixerPipelineConfig[ListTweetsQuery, Timeline, urt.TimelineResponse] {

override val identifier: MixerPipelineIdentifier = MixerPipelineIdentifier("ListTweets")

private val conversationServiceCandidatePipelineConfig =

conversationServiceCandidatePipelineConfigBuilder.build(

Seq(

NonEmptyCandidatesGate(

SpecificPipelines(listTweetsTimelineServiceCandidatePipelineConfig.identifier))

),

ListConversationServiceCandidateDecorator()

)

private val listTweetsAdsCandidatePipelineConfig = listTweetsAdsCandidatePipelineBuilder.build(

SpecificPipelines(listTweetsTimelineServiceCandidatePipelineConfig.identifier)

)

override val candidatePipelines: Seq[CandidatePipelineConfig[ListTweetsQuery, \_, \_, \_]] =

Seq(listTweetsTimelineServiceCandidatePipelineConfig)

override val dependentCandidatePipelines: Seq[

DependentCandidatePipelineConfig[ListTweetsQuery, \_, \_, \_]

] =

Seq(conversationServiceCandidatePipelineConfig, listTweetsAdsCandidatePipelineConfig)

override val failOpenPolicies: Map[CandidatePipelineIdentifier, FailOpenPolicy] = Map(

conversationServiceCandidatePipelineConfig.identifier -> FailOpenPolicy.Always,

listTweetsAdsCandidatePipelineConfig.identifier -> FailOpenPolicy.Always)

override val resultSelectors: Seq[Selector[ListTweetsQuery]] = Seq(

UpdateSortCandidates(

ordering = CandidatesUtil.reverseChronTweetsOrdering,

candidatePipeline = conversationServiceCandidatePipelineConfig.identifier

),

InsertAppendResults(candidatePipeline = conversationServiceCandidatePipelineConfig.identifier),

InsertAdResults(

surfaceAreaName = AdsInjectionSurfaceAreas.HomeTimeline,

adsInjector = adsInjector.forSurfaceArea(AdsInjectionSurfaceAreas.HomeTimeline),

adsCandidatePipeline = listTweetsAdsCandidatePipelineConfig.identifier

),

)

override val fetchQueryFeatures: Seq[QueryFeatureHydrator[ListTweetsQuery]] = Seq(

requestQueryFeatureHydrator,

sgsFollowedUsersQueryFeatureHydrator

)

private val homeScribeClientEventSideEffect = HomeScribeClientEventSideEffect(

enableScribeClientEvents = enableScribeClientEvents,

logPipelinePublisher = clientEventsScribeEventPublisher,

injectedTweetsCandidatePipelineIdentifiers =

Seq(conversationServiceCandidatePipelineConfig.identifier),

adsCandidatePipelineIdentifier = Some(listTweetsAdsCandidatePipelineConfig.identifier),

)

override val resultSideEffects: Seq[PipelineResultSideEffect[ListTweetsQuery, Timeline]] =

Seq(homeScribeClientEventSideEffect)

override val domainMarshaller: DomainMarshaller[ListTweetsQuery, Timeline] = {

val instructionBuilders = Seq(

ReplaceEntryInstructionBuilder(ReplaceAllEntries),

AddEntriesWithReplaceAndShowAlertInstructionBuilder(),

ShowAlertInstructionBuilder()

)

val idSelector: PartialFunction[UniversalNoun[\_], Long] = {

// exclude ads while determining tweet cursor values

case item: TweetItem if item.promotedMetadata.isEmpty => item.id

case module: TimelineModule

if module.items.headOption.exists(\_.item.isInstanceOf[TweetItem]) =>

module.items.last.item match {

case item: TweetItem => item.id

}

}

val topCursorBuilder = OrderedTopCursorBuilder(idSelector)

val bottomCursorBuilder =

OrderedBottomCursorBuilder(idSelector, GapIncludeInstruction.inverse())

val gapCursorBuilder = OrderedGapCursorBuilder(idSelector, GapIncludeInstruction)

val metadataBuilder = UrtMetadataBuilder(

title = None,

scribeConfigBuilder = Some(

StaticTimelineScribeConfigBuilder(

TimelineScribeConfig(page = Some("list\_tweets"), section = None, entityToken = None)))

)

UrtDomainMarshaller(

instructionBuilders = instructionBuilders,

metadataBuilder = Some(metadataBuilder),

cursorBuilders = Seq(topCursorBuilder, bottomCursorBuilder, gapCursorBuilder)

)

}

override val transportMarshaller: TransportMarshaller[Timeline, urt.TimelineResponse] =

urtTransportMarshaller

}