package com.twitter.product\_mixer.component\_library.decorator.urt.builder.metadata

import com.twitter.bijection.scrooge.BinaryScalaCodec

import com.twitter.bijection.Base64String

import com.twitter.bijection.{Injection => Serializer}

import com.twitter.product\_mixer.component\_library.decorator.urt.builder.metadata.ConversationTweetClientEventDetailsBuilder.ControllerDataSerializer

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

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.metadata.BaseClientEventDetailsBuilder

import com.twitter.product\_mixer.component\_library.model.candidate.BaseTweetCandidate

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

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

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

import com.twitter.suggests.controller\_data.home\_tweets.thriftscala.HomeTweetsControllerData

import com.twitter.suggests.controller\_data.thriftscala.ControllerData

import com.twitter.suggests.controller\_data.home\_tweets.v1.thriftscala.{

HomeTweetsControllerData => HomeTweetsControllerDataV1

}

import com.twitter.suggests.controller\_data.v2.thriftscala.{ControllerData => ControllerDataV2}

object ConversationTweetClientEventDetailsBuilder {

implicit val ByteSerializer: Serializer[ControllerData, Array[Byte]] =

BinaryScalaCodec(ControllerData)

val ControllerDataSerializer: Serializer[ControllerData, String] =

Serializer.connect[ControllerData, Array[Byte], Base64String, String]

}

case class ConversationTweetClientEventDetailsBuilder[-Query <: PipelineQuery](

injectionType: Option[String])

extends BaseClientEventDetailsBuilder[Query, BaseTweetCandidate] {

override def apply(

query: Query,

tweetCandidate: BaseTweetCandidate,

candidateFeatures: FeatureMap

): Option[ClientEventDetails] =

Some(

ClientEventDetails(

conversationDetails = None,

timelinesDetails = Some(

TimelinesDetails(

injectionType = injectionType,

controllerData = Some(buildControllerData(query.getUserOrGuestId)),

sourceData = None)),

articleDetails = None,

liveEventDetails = None,

commerceDetails = None

))

private def buildControllerData(traceId: Option[Long]): String =

ControllerDataSerializer(

ControllerData.V2(

ControllerDataV2.HomeTweets(

HomeTweetsControllerData.V1(

HomeTweetsControllerDataV1(

tweetTypesBitmap = 0L,

traceId = traceId,

)

)

)

)

)

}