package com.twitter.product\_mixer.component\_library.pipeline.candidate.ads

import com.twitter.adserver.thriftscala.AdImpression

import com.twitter.product\_mixer.component\_library.model.candidate.ads.AdsCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.ads.AdsTweetCandidate

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineResultsTransformer

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.PipelineFailure

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.UnexpectedCandidateResult

object AdsCandidatePipelineResultsTransformer

extends CandidatePipelineResultsTransformer[AdImpression, AdsCandidate] {

override def transform(sourceResult: AdImpression): AdsCandidate =

(sourceResult.nativeRtbCreative, sourceResult.promotedTweetId) match {

case (None, Some(promotedTweetId)) =>

AdsTweetCandidate(

id = promotedTweetId,

adImpression = sourceResult

)

case (Some(\_), None) =>

throw unsupportedAdImpressionPipelineFailure(

impression = sourceResult,

reason = "Received ad impression with rtbCreative")

case (Some(\_), Some(\_)) =>

throw unsupportedAdImpressionPipelineFailure(

impression = sourceResult,

reason = "Received ad impression with both rtbCreative and promoted tweetId")

case (None, None) =>

throw unsupportedAdImpressionPipelineFailure(

impression = sourceResult,

reason = "Received ad impression with neither rtbCreative nor promoted tweetId")

}

private def unsupportedAdImpressionPipelineFailure(impression: AdImpression, reason: String) =

PipelineFailure(

UnexpectedCandidateResult,

reason =

s"Unsupported AdImpression ($reason). impressionString: ${impression.impressionString}")

}