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

import com.twitter.product\_mixer.component\_library.decorator.urt.builder.stringcenter.Str

import com.twitter.product\_mixer.component\_library.model.candidate.trends\_events.PromotedTrendAdvertiserNameFeature

import com.twitter.product\_mixer.component\_library.model.candidate.trends\_events.TrendTweetCount

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

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

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

import com.twitter.trends.trending\_content.util.CompactingNumberLocalizer

case class TrendMetaDescriptionBuilder[-Query <: PipelineQuery, -Candidate <: UniversalNoun[Any]](

promotedByMetaDescriptionStr: Str[PipelineQuery, UniversalNoun[Any]],

tweetCountMetaDescriptionStr: Str[PipelineQuery, UniversalNoun[Any]],

compactingNumberLocalizer: CompactingNumberLocalizer) {

def apply(

query: Query,

candidate: Candidate,

candidateFeatures: FeatureMap

): Option[String] = {

val promotedMetaDescription =

candidateFeatures.getOrElse(PromotedTrendAdvertiserNameFeature, None).map { advertiserName =>

promotedByMetaDescriptionStr(query, candidate, candidateFeatures).format(advertiserName)

}

val organicMetaDescription = candidateFeatures.getOrElse(TrendTweetCount, None).map {

tweetCount =>

val compactedTweetCount = compactingNumberLocalizer.localizeAndCompact(

query.getLanguageCode

.getOrElse("en"),

tweetCount)

tweetCountMetaDescriptionStr(query, candidate, candidateFeatures).format(

compactedTweetCount)

}

promotedMetaDescription.orElse(organicMetaDescription)

}

}