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

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

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

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.stringcenter.BaseStringCenterPlaceholderBuilder

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

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

import com.twitter.stringcenter.client.StringCenter

import com.twitter.stringcenter.client.core.ExternalString

case class StrStatic(

text: String)

extends BaseStr[PipelineQuery, UniversalNoun[Any]] {

def apply(

query: PipelineQuery,

candidate: UniversalNoun[Any],

candidateFeatures: FeatureMap

): String = text

}

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

text: ExternalString,

stringCenter: StringCenter,

stringCenterPlaceholderBuilder: Option[BaseStringCenterPlaceholderBuilder[Query, Candidate]] =

None)

extends BaseStr[Query, Candidate] {

def apply(query: Query, candidate: Candidate, candidateFeatures: FeatureMap): String = {

val placeholderMapOpt =

stringCenterPlaceholderBuilder.map(\_.apply(query, candidate, candidateFeatures))

stringCenter.prepare(

externalString = text,

placeholders = placeholderMapOpt.getOrElse(Map.empty[String, Any])

)

}

}