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

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

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

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

import com.twitter.timelines.configapi.FSEnumParam

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.topic.BasicTopicDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.topic.PillTopicDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.topic.NoIconTopicDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.topic.PillWithoutActionIconDisplayType

import com.twitter.product\_mixer.core.model.marshalling.response.urt.item.topic.TopicDisplayType

import com.twitter.product\_mixer.core.functional\_component.decorator.urt.builder.item.topic.BaseTopicDisplayTypeBuilder

object TopicCandidateDisplayType extends Enumeration {

type TopicDisplayType = Value

val Basic = Value

val Pill = Value

val NoIcon = Value

val PillWithoutActionIcon = Value

}

case class ParamTopicDisplayTypeBuilder(

displayTypeParam: FSEnumParam[TopicCandidateDisplayType.type])

extends BaseTopicDisplayTypeBuilder[PipelineQuery, TopicCandidate] {

override def apply(

query: PipelineQuery,

candidate: TopicCandidate,

candidateFeatures: FeatureMap

): Option[TopicDisplayType] = {

val displayType = query.params(displayTypeParam)

displayType match {

case TopicCandidateDisplayType.Basic => Some(BasicTopicDisplayType)

case TopicCandidateDisplayType.Pill => Some(PillTopicDisplayType)

case TopicCandidateDisplayType.NoIcon =>

Some(NoIconTopicDisplayType)

case TopicCandidateDisplayType.PillWithoutActionIcon => Some(PillWithoutActionIconDisplayType)

}

}

}