package com.twitter.product\_mixer.component\_library.model.cursor

import com.twitter.product\_mixer.core.feature.Feature

import com.twitter.product\_mixer.core.model.marshalling.response.slice.CursorType

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

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

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

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

import com.twitter.product\_mixer.core.model.marshalling.response.urt.operation.{

CursorType => UrtCursorType

}

case object PreviousCursorFeature

extends Feature[PipelineQuery with HasPipelineCursor[UrtPassThroughCursor], String]

case object NextCursorFeature

extends Feature[PipelineQuery with HasPipelineCursor[UrtPassThroughCursor], String]

/\*\*

\* Cursor model that may be used when we want to pass through the cursor value from and back to

\* a downstream as-is.

\*

\* @param initialSortIndex See [[UrtPipelineCursor]]

\* @param cursorValue the pass through cursor

\*/

case class UrtPassThroughCursor(

override val initialSortIndex: Long,

cursorValue: String,

cursorType: Option[UrtCursorType] = None)

extends UrtPipelineCursor

case class PassThroughCursor(

cursorValue: String,

cursorType: Option[CursorType] = None)

extends PipelineCursor