package com.twitter.product\_mixer.component\_library.premarshaller.urt.builder

import com.twitter.product\_mixer.component\_library.model.cursor.UrtOrderedCursor

import com.twitter.product\_mixer.component\_library.premarshaller.cursor.UrtCursorSerializer

import com.twitter.product\_mixer.component\_library.premarshaller.urt.builder.OrderedTopCursorBuilder.TopCursorOffset

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

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

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

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

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

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

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

case object OrderedTopCursorBuilder {

// Ensure that the next initial sort index is at least 10000 entries away from top cursor's

// current sort index. This is to ensure that the contents of the next page can be populated

// without being assigned sort indices which conflict with that of the current page. This assumes

// that each page will have fewer than 10000 entries.

val TopCursorOffset = 10000L

}

/\*\*

\* Builds [[UrtOrderedCursor]] in the Top position

\*

\* @param idSelector Specifies the entry from which to derive the `id` field

\* @param serializer Converts the cursor to an encoded string

\*/

case class OrderedTopCursorBuilder(

idSelector: PartialFunction[UniversalNoun[\_], Long],

serializer: PipelineCursorSerializer[UrtOrderedCursor] = UrtCursorSerializer)

extends UrtCursorBuilder[

PipelineQuery with HasPipelineCursor[UrtOrderedCursor]

] {

override val cursorType: CursorType = TopCursor

override def cursorValue(

query: PipelineQuery with HasPipelineCursor[UrtOrderedCursor],

timelineEntries: Seq[TimelineEntry]

): String = {

val topId = timelineEntries.collectFirst(idSelector)

val id = topId.orElse(query.pipelineCursor.flatMap(\_.id))

val cursor = UrtOrderedCursor(

initialSortIndex = cursorSortIndex(query, timelineEntries) + TopCursorOffset,

id = id,

cursorType = Some(cursorType)

)

serializer.serializeCursor(cursor)

}

}