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

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

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

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

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

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

import com.twitter.product\_mixer.component\_library.premarshaller.cursor.CursorSerializer.CursorThriftSerializer

import com.twitter.product\_mixer.component\_library.{thriftscala => t}

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

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

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

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

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.IllegalStateFailure

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.MalformedCursor

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.PipelineFailure

import com.twitter.search.common.util.bloomfilter.AdaptiveLongIntBloomFilterSerializer

import com.twitter.product\_mixer.core.functional\_component.marshaller.response.urt.operation.CursorTypeMarshaller

/\*\*

\* Handles serialization and deserialization for all supported URT cursors

\*/

object UrtCursorSerializer extends PipelineCursorSerializer[UrtPipelineCursor] {

val SerializedUrtPlaceholderCursor = CursorThriftSerializer.toString(

t.ProductMixerRequestCursor.UrtPlaceholderCursor(t.UrtPlaceholderCursor()))

val cursorTypeMarshaller = new CursorTypeMarshaller()

override def serializeCursor(cursor: UrtPipelineCursor): String =

cursor match {

case UrtOrderedCursor(initialSortIndex, id, cursorType, gapBoundaryId) =>

val thriftCursor = t.ProductMixerRequestCursor.UrtOrderedCursor(

t.UrtOrderedCursor(

initialSortIndex = initialSortIndex,

id = id,

cursorType.map(cursorTypeMarshaller.apply),

gapBoundaryId = gapBoundaryId))

CursorThriftSerializer.toString(thriftCursor)

case UrtUnorderedExcludeIdsCursor(initialSortIndex, excludedIds) =>

val thriftCursor = t.ProductMixerRequestCursor.UrtUnorderedExcludeIdsCursor(

t.UrtUnorderedExcludeIdsCursor(

initialSortIndex = initialSortIndex,

excludedIds = Some(excludedIds)))

CursorThriftSerializer.toString(thriftCursor)

case UrtUnorderedBloomFilterCursor(initialSortIndex, longIntBloomFilter) =>

val thriftCursor = t.ProductMixerRequestCursor.UrtUnorderedBloomFilterCursor(

t.UrtUnorderedBloomFilterCursor(

initialSortIndex = initialSortIndex,

serializedLongIntBloomFilter =

AdaptiveLongIntBloomFilterSerializer.serialize(longIntBloomFilter)

))

CursorThriftSerializer.toString(thriftCursor)

case UrtPassThroughCursor(initialSortIndex, cursorValue, cursorType) =>

val thriftCursor = t.ProductMixerRequestCursor.UrtPassThroughCursor(

t.UrtPassThroughCursor(

initialSortIndex = initialSortIndex,

cursorValue = cursorValue,

cursorType = cursorType.map(cursorTypeMarshaller.apply)

))

CursorThriftSerializer.toString(thriftCursor)

case UrtPlaceholderCursor() =>

SerializedUrtPlaceholderCursor

case \_ =>

throw PipelineFailure(IllegalStateFailure, "Unknown cursor type")

}

def deserializeOrderedCursor(cursorString: String): Option[UrtOrderedCursor] = {

deserializeUrtCursor(

cursorString,

{

case Some(

t.ProductMixerRequestCursor.UrtOrderedCursor(

t.UrtOrderedCursor(initialSortIndex, id, cursorType, gapBoundaryId))) =>

Some(

UrtOrderedCursor(

initialSortIndex = initialSortIndex,

id = id,

cursorType = cursorType.map(cursorTypeMarshaller.unmarshall),

gapBoundaryId))

}

)

}

def deserializeUnorderedExcludeIdsCursor(

cursorString: String

): Option[UrtUnorderedExcludeIdsCursor] = {

deserializeUrtCursor(

cursorString,

{

case Some(

t.ProductMixerRequestCursor.UrtUnorderedExcludeIdsCursor(

t.UrtUnorderedExcludeIdsCursor(initialSortIndex, excludedIdsOpt))) =>

Some(

UrtUnorderedExcludeIdsCursor(

initialSortIndex = initialSortIndex,

excludedIds = excludedIdsOpt.getOrElse(Seq.empty)))

}

)

}

def deserializeUnorderedBloomFilterCursor(

cursorString: String

): Option[UrtUnorderedBloomFilterCursor] = {

deserializeUrtCursor(

cursorString,

{

case Some(

t.ProductMixerRequestCursor.UrtUnorderedBloomFilterCursor(

t.UrtUnorderedBloomFilterCursor(initialSortIndex, serializedLongIntBloomFilter))) =>

val longIntBloomFilter = AdaptiveLongIntBloomFilterSerializer

.deserialize(serializedLongIntBloomFilter).getOrElse(

throw PipelineFailure(

MalformedCursor,

s"Failed to deserialize UrtUnorderedBloomFilterCursor from cursor string: $cursorString")

)

Some(

UrtUnorderedBloomFilterCursor(

initialSortIndex = initialSortIndex,

longIntBloomFilter = longIntBloomFilter))

}

)

}

def deserializePassThroughCursor(cursorString: String): Option[UrtPassThroughCursor] = {

deserializeUrtCursor(

cursorString,

{

case Some(

t.ProductMixerRequestCursor

.UrtPassThroughCursor(

t.UrtPassThroughCursor(initialSortIndex, cursorValue, cursorType))) =>

Some(

UrtPassThroughCursor(

initialSortIndex = initialSortIndex,

cursorValue = cursorValue,

cursorType = cursorType.map(cursorTypeMarshaller.unmarshall)))

}

)

}

private def deserializeUrtCursor[Cursor <: PipelineCursor](

cursorString: String,

deserializePf: PartialFunction[Option[t.ProductMixerRequestCursor], Option[Cursor]]

): Option[Cursor] = {

deserializeCursor[t.ProductMixerRequestCursor, Cursor](

cursorString,

CursorThriftSerializer,

deserializePf orElse {

case Some(t.ProductMixerRequestCursor.UrtPlaceholderCursor(t.UrtPlaceholderCursor())) =>

// Treat submitted placeholder cursor like an initial page load

None

},

)

}

}