package com.twitter.product\_mixer.component\_library.selector

import com.twitter.product\_mixer.core.functional\_component.common.CandidateScope

import com.twitter.product\_mixer.core.functional\_component.selector.Selector

import CandidateScope.PartitionedCandidates

import com.twitter.product\_mixer.core.functional\_component.common.SpecificPipelines

import com.twitter.product\_mixer.core.functional\_component.selector.SelectorResult

import com.twitter.product\_mixer.core.model.common.identifier.CandidatePipelineIdentifier

import com.twitter.product\_mixer.core.model.common.presentation.CandidateWithDetails

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

import com.twitter.timelines.configapi.Param

/\*\*

\* Insert all candidates from a candidate pipeline at a position below, relative to the last

\* selection of the relative to candidate pipeline. If the relative to candidate pipeline does not

\* contain candidates, then the candidates will be inserted with padding relative to position zero.

\* If the current results are a shorter length than the requested padding, then the candidates will

\* be appended to the results.

\*/

case class InsertRelativePositionResults(

candidatePipeline: CandidatePipelineIdentifier,

relativeToCandidatePipeline: CandidatePipelineIdentifier,

paddingAboveParam: Param[Int])

extends Selector[PipelineQuery] {

override val pipelineScope: CandidateScope = SpecificPipelines(candidatePipeline)

override def apply(

query: PipelineQuery,

remainingCandidates: Seq[CandidateWithDetails],

result: Seq[CandidateWithDetails]

): SelectorResult = {

val paddingAbove = query.params(paddingAboveParam)

assert(paddingAbove >= 0, "Padding above must be equal to or greater than zero")

val PartitionedCandidates(selectedCandidates, otherCandidates) =

pipelineScope.partition(remainingCandidates)

val resultUpdated = if (selectedCandidates.nonEmpty) {

// If the relativeToCandidatePipeline has zero candidates, lastIndexWhere will return -1 which

// will start padding from the zero position

val relativePosition = result.lastIndexWhere(\_.source == relativeToCandidatePipeline) + 1

val position = relativePosition + paddingAbove

if (position < result.length) {

val (left, right) = result.splitAt(position)

left ++ selectedCandidates ++ right

} else {

result ++ selectedCandidates

}

} else {

result

}

SelectorResult(remainingCandidates = otherCandidates, result = resultUpdated)

}

}