package com.twitter.product\_mixer.component\_library.candidate\_source.tweetconvosvc

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

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

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

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

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

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

/\*\*

\* Takes a conversation module item and truncates it to be at most the focal tweet, the focal tweet's

\* in reply to tweet and optionally, the root conversation tweet if desired.

\* @param pipelineScope What pipeline scopes to include in this.

\* @param includeRootTweet Whether to include the root tweet at the top of the conversation or not.

\* @tparam Query

\*/

case class DropMaxConversationModuleItemCandidates[-Query <: PipelineQuery](

override val pipelineScope: CandidateScope,

includeRootTweet: Boolean)

extends Selector[Query] {

override def apply(

query: Query,

remainingCandidates: Seq[CandidateWithDetails],

result: Seq[CandidateWithDetails]

): SelectorResult = {

val updatedCandidates = remainingCandidates.collect {

case moduleCandidate: ModuleCandidateWithDetails if pipelineScope.contains(moduleCandidate) =>

updateConversationModule(moduleCandidate, includeRootTweet)

case candidates => candidates

}

SelectorResult(remainingCandidates = updatedCandidates, result = result)

}

private def updateConversationModule(

module: ModuleCandidateWithDetails,

includeRootTweet: Boolean

): ModuleCandidateWithDetails = {

// If the thread is only the root tweet & a focal tweet replying to it, no truncation can be done.

if (module.candidates.length <= 2) {

module

} else {

// If a thread is more 3 or more tweets, we optionally keep the root tweet if desired, and take

// the focal tweet tweet and its direct ancestor (the one it would have replied to) and return

// those.

val tweetCandidates = module.candidates

val replyAndFocalTweet = tweetCandidates.takeRight(2)

val updatedConversation = if (includeRootTweet) {

tweetCandidates.headOption ++ replyAndFocalTweet

} else {

replyAndFocalTweet

}

module.copy(candidates = updatedConversation.toSeq)

}

}

}