package com.twitter.unified\_user\_actions.adapter.client\_event

import com.twitter.clientapp.thriftscala.EventNamespace

import com.twitter.clientapp.thriftscala.Item

import com.twitter.clientapp.thriftscala.ItemType.Topic

import com.twitter.guide.scribing.thriftscala.TopicModuleMetadata

import com.twitter.guide.scribing.thriftscala.TransparentGuideDetails

import com.twitter.suggests.controller\_data.home\_hitl\_topic\_annotation\_prompt.thriftscala.HomeHitlTopicAnnotationPromptControllerData

import com.twitter.suggests.controller\_data.home\_hitl\_topic\_annotation\_prompt.v1.thriftscala.{

HomeHitlTopicAnnotationPromptControllerData => HomeHitlTopicAnnotationPromptControllerDataV1

}

import com.twitter.suggests.controller\_data.home\_topic\_annotation\_prompt.thriftscala.HomeTopicAnnotationPromptControllerData

import com.twitter.suggests.controller\_data.home\_topic\_annotation\_prompt.v1.thriftscala.{

HomeTopicAnnotationPromptControllerData => HomeTopicAnnotationPromptControllerDataV1

}

import com.twitter.suggests.controller\_data.home\_topic\_follow\_prompt.thriftscala.HomeTopicFollowPromptControllerData

import com.twitter.suggests.controller\_data.home\_topic\_follow\_prompt.v1.thriftscala.{

HomeTopicFollowPromptControllerData => HomeTopicFollowPromptControllerDataV1

}

import com.twitter.suggests.controller\_data.home\_tweets.thriftscala.HomeTweetsControllerData

import com.twitter.suggests.controller\_data.home\_tweets.v1.thriftscala.{

HomeTweetsControllerData => HomeTweetsControllerDataV1

}

import com.twitter.suggests.controller\_data.search\_response.item\_types.thriftscala.ItemTypesControllerData

import com.twitter.suggests.controller\_data.search\_response.thriftscala.SearchResponseControllerData

import com.twitter.suggests.controller\_data.search\_response.topic\_follow\_prompt.thriftscala.SearchTopicFollowPromptControllerData

import com.twitter.suggests.controller\_data.search\_response.tweet\_types.thriftscala.TweetTypesControllerData

import com.twitter.suggests.controller\_data.search\_response.v1.thriftscala.{

SearchResponseControllerData => SearchResponseControllerDataV1

}

import com.twitter.suggests.controller\_data.thriftscala.ControllerData

import com.twitter.suggests.controller\_data.timelines\_topic.thriftscala.TimelinesTopicControllerData

import com.twitter.suggests.controller\_data.timelines\_topic.v1.thriftscala.{

TimelinesTopicControllerData => TimelinesTopicControllerDataV1

}

import com.twitter.suggests.controller\_data.v2.thriftscala.{ControllerData => ControllerDataV2}

import com.twitter.util.Try

object TopicIdUtils {

val DomainId: Long = 131 // Topical Domain

def getTopicId(

item: Item,

namespace: EventNamespace

): Option[Long] =

getTopicIdFromHomeSearch(item)

.orElse(getTopicFromGuide(item))

.orElse(getTopicFromOnboarding(item, namespace))

.orElse(getTopicIdFromItem(item))

def getTopicIdFromItem(item: Item): Option[Long] =

if (item.itemType.contains(Topic))

item.id

else None

def getTopicIdFromHomeSearch(

item: Item

): Option[Long] = {

val decodedControllerData = item.suggestionDetails.flatMap(\_.decodedControllerData)

decodedControllerData match {

case Some(

ControllerData.V2(

ControllerDataV2.HomeTweets(

HomeTweetsControllerData.V1(homeTweets: HomeTweetsControllerDataV1)))

) =>

homeTweets.topicId

case Some(

ControllerData.V2(

ControllerDataV2.HomeTopicFollowPrompt(

HomeTopicFollowPromptControllerData.V1(

homeTopicFollowPrompt: HomeTopicFollowPromptControllerDataV1)))

) =>

homeTopicFollowPrompt.topicId

case Some(

ControllerData.V2(

ControllerDataV2.TimelinesTopic(

TimelinesTopicControllerData.V1(

timelinesTopic: TimelinesTopicControllerDataV1

)))

) =>

Some(timelinesTopic.topicId)

case Some(

ControllerData.V2(

ControllerDataV2.SearchResponse(

SearchResponseControllerData.V1(s: SearchResponseControllerDataV1)))

) =>

s.itemTypesControllerData match {

case Some(

ItemTypesControllerData.TopicFollowControllerData(

topicFollowControllerData: SearchTopicFollowPromptControllerData)) =>

topicFollowControllerData.topicId

case Some(

ItemTypesControllerData.TweetTypesControllerData(

tweetTypesControllerData: TweetTypesControllerData)) =>

tweetTypesControllerData.topicId

case \_ => None

}

case Some(

ControllerData.V2(

ControllerDataV2.HomeTopicAnnotationPrompt(

HomeTopicAnnotationPromptControllerData.V1(

homeTopicAnnotationPrompt: HomeTopicAnnotationPromptControllerDataV1

)))

) =>

Some(homeTopicAnnotationPrompt.topicId)

case Some(

ControllerData.V2(

ControllerDataV2.HomeHitlTopicAnnotationPrompt(

HomeHitlTopicAnnotationPromptControllerData.V1(

homeHitlTopicAnnotationPrompt: HomeHitlTopicAnnotationPromptControllerDataV1

)))

) =>

Some(homeHitlTopicAnnotationPrompt.topicId)

case \_ => None

}

}

def getTopicFromOnboarding(

item: Item,

namespace: EventNamespace

): Option[Long] =

if (namespace.page.contains("onboarding") &&

(namespace.section.exists(\_.contains("topic")) ||

namespace.component.exists(\_.contains("topic")) ||

namespace.element.exists(\_.contains("topic")))) {

item.description.flatMap { description =>

// description: "id=123,main=xyz,row=1"

val tokens = description.split(",").headOption.map(\_.split("="))

tokens match {

case Some(Array("id", token, \_\*)) => Try(token.toLong).toOption

case \_ => None

}

}

} else None

def getTopicFromGuide(

item: Item

): Option[Long] =

item.guideItemDetails.flatMap {

\_.transparentGuideDetails match {

case Some(TransparentGuideDetails.TopicMetadata(topicMetadata)) =>

topicMetadata match {

case TopicModuleMetadata.TttInterest(\_) =>

None

case TopicModuleMetadata.SemanticCoreInterest(semanticCoreInterest) =>

if (semanticCoreInterest.domainId == DomainId.toString)

Try(semanticCoreInterest.entityId.toLong).toOption

else None

case TopicModuleMetadata.SimClusterInterest(\_) =>

None

case TopicModuleMetadata.UnknownUnionField(\_) => None

}

case \_ => None

}

}

}