package com.twitter.product\_mixer.component\_library.pipeline.candidate.flexible\_injection\_pipeline.transformer

import com.twitter.onboarding.injections.{thriftscala => flipinjection}

import com.twitter.product\_mixer.component\_library.candidate\_source.flexible\_injection\_pipeline.IntermediatePrompt

import com.twitter.product\_mixer.component\_library.model.candidate.BasePromptCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.FullCoverPromptCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.HalfCoverPromptCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.InlinePromptCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.PromptCarouselTileCandidate

import com.twitter.product\_mixer.component\_library.model.candidate.RelevancePromptCandidate

import com.twitter.product\_mixer.core.functional\_component.transformer.CandidatePipelineResultsTransformer

import com.twitter.product\_mixer.core.functional\_component.marshaller.TransportMarshaller

object PromptResultsTransformer

extends CandidatePipelineResultsTransformer[

IntermediatePrompt,

BasePromptCandidate[Any]

] {

/\*\*

\* Transforms a Flip Injection to a Product Mixer domain object deriving from BasePromptCandidate.

\* Supported injection types have to match those declared in com.twitter.product\_mixer.component\_library.transformer.flexible\_injection\_pipeline.FlipQueryTransformer#supportedPromptFormats

\*/

override def transform(input: IntermediatePrompt): BasePromptCandidate[Any] =

input.injection match {

case inlinePrompt: flipinjection.Injection.InlinePrompt =>

InlinePromptCandidate(id = inlinePrompt.inlinePrompt.injectionIdentifier

.getOrElse(throw new MissingInjectionId(input.injection)))

case \_: flipinjection.Injection.FullCover =>

FullCoverPromptCandidate(id = "0")

case \_: flipinjection.Injection.HalfCover =>

HalfCoverPromptCandidate(id = "0")

case \_: flipinjection.Injection.TilesCarousel =>

PromptCarouselTileCandidate(id =

input.offsetInModule.getOrElse(throw FlipPromptOffsetInModuleMissing))

case relevancePrompt: flipinjection.Injection.RelevancePrompt =>

RelevancePromptCandidate(

id = relevancePrompt.relevancePrompt.injectionIdentifier,

position = relevancePrompt.relevancePrompt.requestedPosition.map(\_.toInt))

case injection => throw new UnsupportedInjectionType(injection)

}

}

class MissingInjectionId(injection: flipinjection.Injection)

extends IllegalArgumentException(

s"Injection identifier is missing ${TransportMarshaller.getSimpleName(injection.getClass)}")

class UnsupportedInjectionType(injection: flipinjection.Injection)

extends UnsupportedOperationException(

s"Unsupported FLIP injection Type : ${TransportMarshaller.getSimpleName(injection.getClass)}")

object FlipPromptOffsetInModuleMissing

extends NoSuchElementException(

"FlipPromptOffsetInModuleFeature must be set for the TilesCarousel FLIP injection in PromptCandidateSource")