package com.twitter.product\_mixer.component\_library.decorator.urt.builder.flexible\_injection\_pipeline

import com.twitter.product\_mixer.component\_library.decorator.urt.GroupByKey

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

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

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.model.common.UniversalNoun

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

object FlipPromptModuleGrouping extends GroupByKey[PipelineQuery, UniversalNoun[Any], Int] {

override def apply(

query: PipelineQuery,

candidate: UniversalNoun[Any],

candidateFeatures: FeatureMap

): Option[Int] = {

val injection = candidateFeatures.get(FlipPromptInjectionsFeature)

val offsetInModule = candidateFeatures.getOrElse(FlipPromptOffsetInModuleFeature, None)

// We return None for any candidate that doesn't have an offsetInModule, so that they are left as independent items.

// Otherwise, we return a hash of the injection instance which will be used to aggregate candidates with matching values into a module.

offsetInModule.map(\_ => injection.hashCode())

}

}