package com.twitter.follow\_recommendations.products.common

import com.twitter.follow\_recommendations.assembler.models.Layout

import com.twitter.follow\_recommendations.common.base.BaseRecommendationFlow

import com.twitter.follow\_recommendations.common.base.Transform

import com.twitter.follow\_recommendations.common.models.DisplayLocation

import com.twitter.follow\_recommendations.common.models.Recommendation

import com.twitter.follow\_recommendations.models.RecommendationRequest

import com.twitter.product\_mixer.core.model.marshalling.request.{Product => ProductMixerProduct}

import com.twitter.stitch.Stitch

import com.twitter.timelines.configapi.Params

trait Product {

/\*\* Each product also requires a human-readable name.

\* You can change this at any time

\*/

def name: String

/\*\*

\* Every product needs a machine-friendly identifier for internal use.

\* You should use the same name as the product package name.

\* Except dashes are better than underscore

\*

\* Avoid changing this once it's in production.

\*/

def identifier: String

def displayLocation: DisplayLocation

def selectWorkflows(

request: ProductRequest

): Stitch[Seq[BaseRecommendationFlow[ProductRequest, \_ <: Recommendation]]]

/\*\*

\* Blender is responsible for blending together the candidates generated by different flows used

\* in a product. For example, if a product uses two flows, it is blender's responsibility to

\* interleave their generated candidates together and make a unified sequence of candidates.

\*/

def blender: Transform[ProductRequest, Recommendation]

/\*\*

\* It is resultsTransformer job to do any final transformations needed on the final list of

\* candidates generated by a product. For example, if a final quality check on candidates needed,

\* resultsTransformer will handle it.

\*/

def resultsTransformer(request: ProductRequest): Stitch[Transform[ProductRequest, Recommendation]]

def enabled(request: ProductRequest): Stitch[Boolean]

def layout: Option[Layout] = None

def productMixerProduct: Option[ProductMixerProduct] = None

}

case class ProductRequest(recommendationRequest: RecommendationRequest, params: Params)