package com.twitter.product\_mixer.component\_library.feature\_hydrator.candidate.param\_gated

import com.twitter.product\_mixer.component\_library.feature\_hydrator.candidate.param\_gated.ParamGatedCandidateFeatureHydrator.IdentifierPrefix

import com.twitter.product\_mixer.core.feature.Feature

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

import com.twitter.product\_mixer.core.functional\_component.common.alert.Alert

import com.twitter.product\_mixer.core.functional\_component.feature\_hydrator.CandidateFeatureHydrator

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

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

import com.twitter.product\_mixer.core.model.common.identifier.FeatureHydratorIdentifier

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

import com.twitter.stitch.Stitch

import com.twitter.timelines.configapi.Param

/\*\*

\* A [[CandidateFeatureHydrator]] with [[Conditionally]] based on a [[Param]]

\*

\* @param enabledParam the param to turn this [[CandidateFeatureHydrator]] on and off

\* @param candidateFeatureHydrator the underlying [[CandidateFeatureHydrator]] to run when `enabledParam` is true

\* @tparam Query The domain model for the query or request

\* @tparam Result The type of the candidates

\*/

case class ParamGatedCandidateFeatureHydrator[

-Query <: PipelineQuery,

-Result <: UniversalNoun[Any]

](

enabledParam: Param[Boolean],

candidateFeatureHydrator: CandidateFeatureHydrator[Query, Result])

extends CandidateFeatureHydrator[Query, Result]

with Conditionally[Query] {

override val identifier: FeatureHydratorIdentifier = FeatureHydratorIdentifier(

IdentifierPrefix + candidateFeatureHydrator.identifier.name)

override val alerts: Seq[Alert] = candidateFeatureHydrator.alerts

override val features: Set[Feature[\_, \_]] = candidateFeatureHydrator.features

override def onlyIf(query: Query): Boolean =

Conditionally.and(query, candidateFeatureHydrator, query.params(enabledParam))

override def apply(

query: Query,

candidate: Result,

existingFeatures: FeatureMap

): Stitch[FeatureMap] = candidateFeatureHydrator.apply(query, candidate, existingFeatures)

}

object ParamGatedCandidateFeatureHydrator {

val IdentifierPrefix = "ParamGated"

}