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

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

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

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

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

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

import com.twitter.stitch.Stitch

import com.twitter.timelines.configapi.Param

/\*\*

\* This predicate allows us to filter candidates given its source.

\* To avoid bucket dilution, we only want to evaluate the param (which would implicitly trigger

\* bucketing for FSParams) only if the candidate source fn yields true.

\* The param provided should be true when we want to keep the candidate and false otherwise.

\*/

class CandidateSourceParamPredicate(

val param: Param[Boolean],

val reason: FilterReason,

candidateSources: Set[CandidateSourceIdentifier])

extends Predicate[CandidateUser] {

override def apply(candidate: CandidateUser): Stitch[PredicateResult] = {

// we want to avoid evaluating the param if the candidate source fn yields false

if (candidate.getCandidateSources.keys.exists(candidateSources.contains) && !candidate.params(

param)) {

Stitch.value(PredicateResult.Invalid(Set(reason)))

} else {

Stitch.value(PredicateResult.Valid)

}

}

}