package com.twitter.frigate.pushservice

import com.twitter.frigate.common.base.Candidate

import com.twitter.frigate.common.base.SocialGraphServiceRelationshipMap

import com.twitter.frigate.common.base.TweetAuthor

import com.twitter.frigate.common.rec\_types.RecTypes.isInNetworkTweetType

import com.twitter.frigate.pushservice.model.PushTypes.PushCandidate

import com.twitter.hermit.predicate.Predicate

package object predicate {

implicit class CandidatesWithAuthorFollowPredicates(

predicate: Predicate[

PushCandidate with TweetAuthor with SocialGraphServiceRelationshipMap

]) {

def applyOnlyToAuthorBeingFollowPredicates: Predicate[Candidate] =

predicate.optionalOn[Candidate](

{

case candidate: PushCandidate with TweetAuthor with SocialGraphServiceRelationshipMap

if isInNetworkTweetType(candidate.commonRecType) =>

Some(candidate)

case \_ =>

None

},

missingResult = true

)

}

implicit class TweetCandidateWithTweetAuthor(

predicate: Predicate[

PushCandidate with TweetAuthor with SocialGraphServiceRelationshipMap

]) {

def applyOnlyToBasicTweetPredicates: Predicate[Candidate] =

predicate.optionalOn[Candidate](

{

case candidate: PushCandidate with TweetAuthor with SocialGraphServiceRelationshipMap

if isInNetworkTweetType(candidate.commonRecType) =>

Some(candidate)

case \_ =>

None

},

missingResult = true

)

}

}