package com.twitter.tweetypie.util

import com.twitter.escherbird.thriftscala.TweetEntityAnnotation

import com.twitter.tweetypie.thriftscala.EscherbirdEntityAnnotations

import com.twitter.tweetypie.thriftscala.Tweet

object CommunityAnnotation {

val groupId: Long = 8

val domainId: Long = 31

def apply(communityId: Long): TweetEntityAnnotation =

TweetEntityAnnotation(groupId, domainId, entityId = communityId)

def unapply(annotation: TweetEntityAnnotation): Option[Long] =

annotation match {

case TweetEntityAnnotation(`groupId`, `domainId`, entityId) => Some(entityId)

case \_ => None

}

// Returns None instead of Some(Seq()) when there are non-community annotations present

def additionalFieldsToCommunityIDs(additionalFields: Tweet): Option[Seq[Long]] = {

additionalFields.escherbirdEntityAnnotations

.map {

case EscherbirdEntityAnnotations(entityAnnotations) =>

entityAnnotations.flatMap(CommunityAnnotation.unapply)

}.filter(\_.nonEmpty)

}

}