package com.twitter.unified\_user\_actions.adapter.tweetypie\_event

import com.twitter.tweetypie.thriftscala.EditControl

import com.twitter.tweetypie.thriftscala.EditControlEdit

import com.twitter.tweetypie.thriftscala.Tweet

sealed trait TweetypieTweetType

object TweetTypeDefault extends TweetypieTweetType

object TweetTypeReply extends TweetypieTweetType

object TweetTypeRetweet extends TweetypieTweetType

object TweetTypeQuote extends TweetypieTweetType

object TweetTypeEdit extends TweetypieTweetType

object TweetypieEventUtils {

def editedTweetIdFromTweet(tweet: Tweet): Option[Long] = tweet.editControl.flatMap {

case EditControl.Edit(EditControlEdit(initialTweetId, \_)) => Some(initialTweetId)

case \_ => None

}

def tweetTypeFromTweet(tweet: Tweet): Option[TweetypieTweetType] = {

val data = tweet.coreData

val inReplyingToStatusIdOpt = data.flatMap(\_.reply).flatMap(\_.inReplyToStatusId)

val shareOpt = data.flatMap(\_.share)

val quotedTweetOpt = tweet.quotedTweet

val editedTweetIdOpt = editedTweetIdFromTweet(tweet)

(inReplyingToStatusIdOpt, shareOpt, quotedTweetOpt, editedTweetIdOpt) match {

// Reply

case (Some(\_), None, \_, None) =>

Some(TweetTypeReply)

// For any kind of retweet (be it retweet of quote tweet or retweet of a regular tweet)

// we only need to look at the `share` field

// https://confluence.twitter.biz/pages/viewpage.action?spaceKey=CSVC&title=TweetyPie+FAQ#TweetypieFAQ-HowdoItellifaTweetisaRetweet

case (None, Some(\_), \_, None) =>

Some(TweetTypeRetweet)

// quote

case (None, None, Some(\_), None) =>

Some(TweetTypeQuote)

// create

case (None, None, None, None) =>

Some(TweetTypeDefault)

// edit

case (None, None, \_, Some(\_)) =>

Some(TweetTypeEdit)

// reply and retweet shouldn't be present at the same time

case (Some(\_), Some(\_), \_, \_) =>

None

// reply and edit / retweet and edit shouldn't be present at the same time

case (Some(\_), None, \_, Some(\_)) | (None, Some(\_), \_, Some(\_)) =>

None

}

}

}