package com.twitter.visibility.interfaces.tweets

import com.twitter.finagle.context.Contexts

import com.twitter.io.Buf

import com.twitter.io.BufByteWriter

import com.twitter.io.ByteReader

import com.twitter.util.Future

import com.twitter.util.Return

import com.twitter.util.Throw

import com.twitter.util.Try

case class TweetypieContext(

isQuotedTweet: Boolean,

isRetweet: Boolean,

hydrateConversationControl: Boolean)

object TweetypieContext {

def let[U](value: TweetypieContext)(f: => Future[U]): Future[U] =

Contexts.broadcast.let(TweetypieContextKey, value)(f)

def get(): Option[TweetypieContext] =

Contexts.broadcast.get(TweetypieContextKey)

}

object TweetypieContextKey

extends Contexts.broadcast.Key[TweetypieContext](

"com.twitter.visibility.interfaces.tweets.TweetypieContext"

) {

override def marshal(value: TweetypieContext): Buf = {

val bw = BufByteWriter.fixed(1)

val byte =

((if (value.isQuotedTweet) 1 else 0) << 0) |

((if (value.isRetweet) 1 else 0) << 1) |

((if (value.hydrateConversationControl) 1 else 0) << 2)

bw.writeByte(byte)

bw.owned()

}

override def tryUnmarshal(buf: Buf): Try[TweetypieContext] = {

if (buf.length != 1) {

Throw(

new IllegalArgumentException(

s"Could not extract Boolean from Buf. Length ${buf.length} but required 1"

)

)

} else {

val byte: Byte = ByteReader(buf).readByte()

Return(

TweetypieContext(

isQuotedTweet = ((byte & 1) == 1),

isRetweet = ((byte & 2) == 2),

hydrateConversationControl = ((byte & 4) == 4)

)

)

}

}

}