package com.twitter.tweetypie.serverutil

import com.twitter.tweetypie.getCashtags

import com.twitter.tweetypie.getHashtags

import com.twitter.tweetypie.getMedia

import com.twitter.tweetypie.getMentions

import com.twitter.tweetypie.getText

import com.twitter.tweetypie.getUrls

import com.twitter.tweetypie.thriftscala.ExtendedTweetMetadata

import com.twitter.tweetypie.thriftscala.ShortenedUrl

import com.twitter.tweetypie.thriftscala.Tweet

import com.twitter.tweetypie.tweettext.Offset

import com.twitter.tweetypie.tweettext.TextEntity

import com.twitter.tweetypie.tweettext.Truncator

import com.twitter.tweetypie.tweettext.TweetText

import com.twitter.tweetypie.thriftscala.entities.Implicits.\_

/\*\*

\* Computes the appropriate truncation index to support rendering on legacy clients.

\*/

object ExtendedTweetMetadataBuilder {

import TweetText.\_

def apply(tweet: Tweet, selfPermalink: ShortenedUrl): ExtendedTweetMetadata = {

def entityRanges[T: TextEntity](entities: Seq[T]): Seq[(Int, Int)] =

entities.map(e => (TextEntity.fromIndex(e).toInt, TextEntity.toIndex(e).toInt))

val allEntityRanges =

Offset.Ranges.fromCodePointPairs(

entityRanges(getUrls(tweet)) ++

entityRanges(getMentions(tweet)) ++

entityRanges(getMedia(tweet)) ++

entityRanges(getHashtags(tweet)) ++

entityRanges(getCashtags(tweet))

)

val text = getText(tweet)

val apiCompatibleTruncationIndex =

// need to leave enough space for ellipsis, space, and self-permalink

Truncator.truncationPoint(

text = text,

maxDisplayLength = OriginalMaxDisplayLength - selfPermalink.shortUrl.length - 2,

atomicUnits = allEntityRanges

)

ExtendedTweetMetadata(

apiCompatibleTruncationIndex = apiCompatibleTruncationIndex.codePointOffset.toInt

)

}

}