package com.twitter.home\_mixer.model.request

import com.twitter.product\_mixer.core.model.marshalling.request.ClientContext

import com.twitter.{timelineservice => tls}

case class DeviceContext(

isPolling: Option[Boolean],

requestContext: Option[String],

latestControlAvailable: Option[Boolean],

autoplayEnabled: Option[Boolean]) {

lazy val requestContextValue: Option[DeviceContext.RequestContext.Value] =

requestContext.flatMap { value =>

val normalizedValue = value.trim.toLowerCase()

DeviceContext.RequestContext.values.find(\_.toString == normalizedValue)

}

def toTimelineServiceDeviceContext(clientContext: ClientContext): tls.DeviceContext =

tls.DeviceContext(

countryCode = clientContext.countryCode,

languageCode = clientContext.languageCode,

clientAppId = clientContext.appId,

ipAddress = clientContext.ipAddress,

guestId = clientContext.guestId,

sessionId = None,

timezone = None,

userAgent = clientContext.userAgent,

deviceId = clientContext.deviceId,

isPolling = isPolling,

requestProvenance = requestContext,

referrer = None,

tfeAuthHeader = None,

mobileDeviceId = clientContext.mobileDeviceId,

isSessionStart = None,

displaySize = None,

isURTRequest = Some(true),

latestControlAvailable = latestControlAvailable,

guestIdMarketing = clientContext.guestIdMarketing,

isInternalOrTwoffice = clientContext.isTwoffice,

browserNotificationPermission = None,

guestIdAds = clientContext.guestIdAds,

)

}

object DeviceContext {

val Empty: DeviceContext = DeviceContext(

isPolling = None,

requestContext = None,

latestControlAvailable = None,

autoplayEnabled = None

)

/\*\*

\* Constants which reflect valid client request provenances (why a request was initiated, encoded

\* by the "request\_context" HTTP parameter).

\*/

object RequestContext extends Enumeration {

val Auto = Value("auto")

val Foreground = Value("foreground")

val Gap = Value("gap")

val Launch = Value("launch")

val ManualRefresh = Value("manual\_refresh")

val Navigate = Value("navigate")

val Polling = Value("polling")

val PullToRefresh = Value("ptr")

val Signup = Value("signup")

val TweetSelfThread = Value("tweet\_self\_thread")

val BackgroundFetch = Value("background\_fetch")

}

}

trait HasDeviceContext {

def deviceContext: Option[DeviceContext]

}