package com.twitter.visibility.models

import com.twitter.visibility.safety\_label\_store.{thriftscala => s}

import com.twitter.visibility.util.NamingUtils

sealed trait MediaSafetyLabelType extends SafetyLabelType {

lazy val name: String = NamingUtils.getFriendlyName(this)

}

object MediaSafetyLabelType extends SafetyLabelType {

val List: List[MediaSafetyLabelType] = s.MediaSafetyLabelType.list.map(fromThrift)

val ActiveLabels: List[MediaSafetyLabelType] = List.filter { labelType =>

labelType != Unknown && labelType != Deprecated

}

private lazy val nameToValueMap: Map[String, MediaSafetyLabelType] =

List.map(l => l.name.toLowerCase -> l).toMap

def fromName(name: String): Option[MediaSafetyLabelType] = nameToValueMap.get(name.toLowerCase)

private val UnknownThriftSafetyLabelType =

s.MediaSafetyLabelType.EnumUnknownMediaSafetyLabelType(UnknownEnumValue)

private lazy val thriftToModelMap: Map[s.MediaSafetyLabelType, MediaSafetyLabelType] = Map(

s.MediaSafetyLabelType.NsfwHighPrecision -> NsfwHighPrecision,

s.MediaSafetyLabelType.NsfwHighRecall -> NsfwHighRecall,

s.MediaSafetyLabelType.NsfwNearPerfect -> NsfwNearPerfect,

s.MediaSafetyLabelType.NsfwCardImage -> NsfwCardImage,

s.MediaSafetyLabelType.Pdna -> Pdna,

s.MediaSafetyLabelType.PdnaNoTreatmentIfVerified -> PdnaNoTreatmentIfVerified,

s.MediaSafetyLabelType.DmcaWithheld -> DmcaWithheld,

s.MediaSafetyLabelType.LegalDemandsWithheld -> LegalDemandsWithheld,

s.MediaSafetyLabelType.LocalLawsWithheld -> LocalLawsWithheld,

s.MediaSafetyLabelType.Reserved10 -> Deprecated,

s.MediaSafetyLabelType.Reserved11 -> Deprecated,

s.MediaSafetyLabelType.Reserved12 -> Deprecated,

s.MediaSafetyLabelType.Reserved13 -> Deprecated,

s.MediaSafetyLabelType.Reserved14 -> Deprecated,

s.MediaSafetyLabelType.Reserved15 -> Deprecated,

s.MediaSafetyLabelType.Reserved16 -> Deprecated,

s.MediaSafetyLabelType.Reserved17 -> Deprecated,

s.MediaSafetyLabelType.Reserved18 -> Deprecated,

s.MediaSafetyLabelType.Reserved19 -> Deprecated,

s.MediaSafetyLabelType.Reserved20 -> Deprecated,

s.MediaSafetyLabelType.Reserved21 -> Deprecated,

s.MediaSafetyLabelType.Reserved22 -> Deprecated,

s.MediaSafetyLabelType.Reserved23 -> Deprecated,

s.MediaSafetyLabelType.Reserved24 -> Deprecated,

s.MediaSafetyLabelType.Reserved25 -> Deprecated,

s.MediaSafetyLabelType.Reserved26 -> Deprecated,

s.MediaSafetyLabelType.Reserved27 -> Deprecated,

)

private lazy val modelToThriftMap: Map[MediaSafetyLabelType, s.MediaSafetyLabelType] =

(for ((k, v) <- thriftToModelMap) yield (v, k)) ++ Map(

Deprecated -> s.MediaSafetyLabelType.EnumUnknownMediaSafetyLabelType(DeprecatedEnumValue),

)

case object NsfwHighPrecision extends MediaSafetyLabelType

case object NsfwHighRecall extends MediaSafetyLabelType

case object NsfwNearPerfect extends MediaSafetyLabelType

case object NsfwCardImage extends MediaSafetyLabelType

case object Pdna extends MediaSafetyLabelType

case object PdnaNoTreatmentIfVerified extends MediaSafetyLabelType

case object DmcaWithheld extends MediaSafetyLabelType

case object LegalDemandsWithheld extends MediaSafetyLabelType

case object LocalLawsWithheld extends MediaSafetyLabelType

case object Deprecated extends MediaSafetyLabelType

case object Unknown extends MediaSafetyLabelType

def fromThrift(safetyLabelType: s.MediaSafetyLabelType): MediaSafetyLabelType =

thriftToModelMap.get(safetyLabelType) match {

case Some(mediaSafetyLabelType) => mediaSafetyLabelType

case \_ =>

safetyLabelType match {

case s.MediaSafetyLabelType.EnumUnknownMediaSafetyLabelType(DeprecatedEnumValue) =>

Deprecated

case \_ =>

Unknown

}

}

def toThrift(safetyLabelType: MediaSafetyLabelType): s.MediaSafetyLabelType = {

modelToThriftMap

.get(safetyLabelType).getOrElse(UnknownThriftSafetyLabelType)

}

}