package com.twitter.visibility.models

sealed trait ViolationLevel extends Product with Serializable {

val level: Int

}

object ViolationLevel {

case object DefaultLevel extends ViolationLevel {

override val level: Int = 0

}

case object Level1 extends ViolationLevel {

override val level: Int = 1

}

case object Level2 extends ViolationLevel {

override val level: Int = 2

}

case object Level3 extends ViolationLevel {

override val level: Int = 3

}

case object Level4 extends ViolationLevel {

override val level: Int = 4

}

private val safetyLabelToViolationLevel: Map[TweetSafetyLabelType, ViolationLevel] = Map(

TweetSafetyLabelType.FosnrHatefulConduct -> Level3,

TweetSafetyLabelType.FosnrHatefulConductLowSeveritySlur -> Level1,

)

val violationLevelToSafetyLabels: Map[ViolationLevel, Set[TweetSafetyLabelType]] =

safetyLabelToViolationLevel.groupBy { case (\_, violationLevel) => violationLevel }.map {

case (violationLevel, collection) => (violationLevel, collection.keySet)

}

def fromTweetSafetyLabel(

tweetSafetyLabel: TweetSafetyLabel

): ViolationLevel = {

safetyLabelToViolationLevel.getOrElse(tweetSafetyLabel.labelType, DefaultLevel)

}

def fromTweetSafetyLabelOpt(

tweetSafetyLabel: TweetSafetyLabel

): Option[ViolationLevel] = {

safetyLabelToViolationLevel.get(tweetSafetyLabel.labelType)

}

}