package com.twitter.visibility.generators

import com.ibm.icu.util.ULocale

import com.twitter.config.yaml.YamlMap

import com.twitter.finagle.stats.StatsReceiver

object CountryNameGenerator {

private val AuroraFilesystemPath = "/usr/local/twitter-config/twitter/config/"

private val ContentBlockingSupportedCountryList = "takedown\_countries.yml"

def providesFromConfigBus(statsReceiver: StatsReceiver): CountryNameGenerator = {

fromFile(AuroraFilesystemPath + ContentBlockingSupportedCountryList, statsReceiver)

}

def providesWithCustomMap(countryCodeMap: Map[String, String], statsReceiver: StatsReceiver) = {

new CountryNameGenerator(countryCodeMap, statsReceiver)

}

private def fromFile(fileName: String, statsReceiver: StatsReceiver) = {

val yamlConfig = YamlMap.load(fileName)

val countryCodeMap: Map[String, String] = yamlConfig.keySet.map { countryCode: String =>

val normalizedCode = countryCode.toUpperCase

val countryName: Option[String] =

yamlConfig.get(Seq(countryCode, "name")).asInstanceOf[Option[String]]

(normalizedCode, countryName.getOrElse(normalizedCode))

}.toMap

new CountryNameGenerator(countryCodeMap, statsReceiver)

}

}

class CountryNameGenerator(countryCodeMap: Map[String, String], statsReceiver: StatsReceiver) {

private val scopedStatsReceiver = statsReceiver.scope("country\_name\_generator")

private val foundCountryReceiver = scopedStatsReceiver.counter("found")

private val missingCountryReceiver = scopedStatsReceiver.counter("missing")

def getCountryName(code: String): String = {

val normalizedCode = code.toUpperCase

countryCodeMap.get(normalizedCode) match {

case Some(retrievedName) => {

foundCountryReceiver.incr()

retrievedName

}

case \_ => {

missingCountryReceiver.incr()

val fallbackName =

new ULocale("", normalizedCode).getDisplayCountry(ULocale.forLanguageTag("en"))

if (fallbackName == "")

normalizedCode

else

fallbackName

}

}

}

}