package com.twitter.follow\_recommendations.common.clients.cache

import com.twitter.bijection.Bijection

import com.twitter.io.Buf

import com.twitter.scrooge.CompactThriftSerializer

import com.twitter.scrooge.ThriftEnum

import com.twitter.scrooge.ThriftStruct

import java.nio.ByteBuffer

abstract class ThriftBijection[T <: ThriftStruct] extends Bijection[Buf, T] {

val serializer: CompactThriftSerializer[T]

override def apply(b: Buf): T = {

val byteArray = Buf.ByteArray.Owned.extract(b)

serializer.fromBytes(byteArray)

}

override def invert(a: T): Buf = {

val byteArray = serializer.toBytes(a)

Buf.ByteArray.Owned(byteArray)

}

}

abstract class ThriftOptionBijection[T <: ThriftStruct] extends Bijection[Buf, Option[T]] {

val serializer: CompactThriftSerializer[T]

override def apply(b: Buf): Option[T] = {

if (b.isEmpty) {

None

} else {

val byteArray = Buf.ByteArray.Owned.extract(b)

Some(serializer.fromBytes(byteArray))

}

}

override def invert(a: Option[T]): Buf = {

a match {

case Some(t) =>

val byteArray = serializer.toBytes(t)

Buf.ByteArray.Owned(byteArray)

case None => Buf.Empty

}

}

}

class ThriftEnumBijection[T <: ThriftEnum](constructor: Int => T) extends Bijection[Buf, T] {

override def apply(b: Buf): T = {

val byteArray = Buf.ByteArray.Owned.extract(b)

val byteBuffer = ByteBuffer.wrap(byteArray)

constructor(byteBuffer.getInt())

}

override def invert(a: T): Buf = {

val byteBuffer: ByteBuffer = ByteBuffer.allocate(4)

byteBuffer.putInt(a.getValue)

Buf.ByteArray.Owned(byteBuffer.array())

}

}

class ThriftEnumOptionBijection[T <: ThriftEnum](constructor: Int => T) extends Bijection[Buf, Option[T]] {

override def apply(b: Buf): Option[T] = {

if (b.isEmpty) {

None

} else {

val byteArray = Buf.ByteArray.Owned.extract(b)

val byteBuffer = ByteBuffer.wrap(byteArray)

Some(constructor(byteBuffer.getInt()))

}

}

override def invert(a: Option[T]): Buf = {

a match {

case Some(obj) => {

val byteBuffer: ByteBuffer = ByteBuffer.allocate(4)

byteBuffer.putInt(obj.getValue)

Buf.ByteArray.Owned(byteBuffer.array())

}

case None => Buf.Empty

}

}

}