package com.twitter.ann.file\_store

import com.twitter.ann.common.thriftscala.FileBasedIndexIdStore

import com.twitter.bijection.Injection

import com.twitter.mediaservices.commons.codec.{ArrayByteBufferCodec, ThriftByteBufferCodec}

import com.twitter.search.common.file.AbstractFile

import com.twitter.storehaus.ReadableStore

import java.nio.ByteBuffer

object ReadableIndexIdFileStore {

/\*\*

\* @param file : File path to read serialized long indexId <-> Id mapping from.

\* @param injection: Injection to convert bytes to Id.

\* @tparam V: Type of Id

\* @return File based Readable Store

\*/

def apply[V](

file: AbstractFile,

injection: Injection[V, Array[Byte]]

): ReadableStore[Long, V] = {

val codec = new ThriftByteBufferCodec(FileBasedIndexIdStore)

val store: Map[Long, V] = codec

.decode(loadFile(file))

.indexIdMap

.getOrElse(Map.empty[Long, ByteBuffer])

.toMap

.mapValues(value => injection.invert(ArrayByteBufferCodec.decode(value)).get)

ReadableStore.fromMap[Long, V](store)

}

private[this] def loadFile(file: AbstractFile): ByteBuffer = {

ArrayByteBufferCodec.encode(file.getByteSource.read())

}

}