package com.twitter.search.core.earlybird.index.inverted;

import java.io.IOException;

import com.twitter.search.common.util.io.flushable.DataDeserializer;

import com.twitter.search.common.util.io.flushable.DataSerializer;

import com.twitter.search.common.util.io.flushable.FlushInfo;

import com.twitter.search.common.util.io.flushable.Flushable;

public class ByteBlockPool extends BaseByteBlockPool implements Flushable {

public ByteBlockPool() {

}

/\*\*

\* Used for loading flushed pool.

\*/

private ByteBlockPool(Pool pool, int bufferUpto, int byteUpTo, int byteOffset) {

super(pool, bufferUpto, byteUpTo, byteOffset);

}

@Override

public FlushHandler getFlushHandler() {

return new FlushHandler(this);

}

public static class FlushHandler extends Flushable.Handler<ByteBlockPool> {

private static final String BUFFER\_UP\_TO\_PROP\_NAME = "bufferUpto";

private static final String BYTE\_UP\_TO\_PROP\_NAME = "byteUpto";

private static final String BYTE\_OFFSET\_PROP\_NAME = "byteOffset";

public FlushHandler(ByteBlockPool objectToFlush) {

super(objectToFlush);

}

public FlushHandler() {

}

@Override

protected void doFlush(FlushInfo flushInfo, DataSerializer out) throws IOException {

ByteBlockPool objectToFlush = getObjectToFlush();

out.writeByteArray2D(objectToFlush.pool.buffers, objectToFlush.bufferUpto + 1);

flushInfo.addIntProperty(BUFFER\_UP\_TO\_PROP\_NAME, objectToFlush.bufferUpto);

flushInfo.addIntProperty(BYTE\_UP\_TO\_PROP\_NAME, objectToFlush.byteUpto);

flushInfo.addIntProperty(BYTE\_OFFSET\_PROP\_NAME, objectToFlush.byteOffset);

}

@Override

protected ByteBlockPool doLoad(FlushInfo flushInfo,

DataDeserializer in) throws IOException {

return new ByteBlockPool(

new BaseByteBlockPool.Pool(in.readByteArray2D()),

flushInfo.getIntProperty(BUFFER\_UP\_TO\_PROP\_NAME),

flushInfo.getIntProperty(BYTE\_UP\_TO\_PROP\_NAME),

flushInfo.getIntProperty(BYTE\_OFFSET\_PROP\_NAME));

}

}

}