#include "internal/error.h"

#include <cstring>

#include <iostream>

#include <twml/BlockFormatWriter.h>

#define WIRE\_TYPE\_LENGTH\_PREFIXED (2)

#define WIRE\_TYPE\_VARINT (0)

#ifndef PATH\_MAX

#define PATH\_MAX (8096)

#endif

#define MARKER\_SIZE (16)

static uint8\_t \_marker[MARKER\_SIZE] = {

0x29, 0xd8, 0xd5, 0x06, 0x58, 0xcd, 0x4c, 0x29,

0xb2, 0xbc, 0x57, 0x99, 0x21, 0x71, 0xbd, 0xff

};

namespace twml {

BlockFormatWriter::BlockFormatWriter(const char \*file\_name, int record\_per\_block) :

file\_name\_(file\_name), record\_index\_(0), records\_per\_block\_(record\_per\_block) {

snprintf(temp\_file\_name\_, PATH\_MAX, "%s.block", file\_name);

outputfile\_ = fopen(file\_name\_, "a");

}

BlockFormatWriter::~BlockFormatWriter() {

fclose(outputfile\_);

}

// TODO: use fstream

int BlockFormatWriter::pack\_tag\_and\_wiretype(FILE \*buffer, uint32\_t tag, uint32\_t wiretype) {

uint8\_t x = ((tag & 0x0f) << 3) | (wiretype & 0x7);

size\_t n = fwrite(&x, 1, 1, buffer);

if (n != 1) {

return -1;

}

return 0;

}

int BlockFormatWriter::pack\_varint\_i32(FILE \*buffer, int value) {

for (int i = 0; i < 10; i++) {

uint8\_t x = value & 0x7F;

value = value >> 7;

if (value != 0) x |= 0x80;

size\_t n = fwrite(&x, 1, 1, buffer);

if (n != 1) {

return -1;

}

if (value == 0) break;

}

return 0;

}

int BlockFormatWriter::pack\_string(FILE \*buffer, const char \*in, size\_t in\_len) {

if (pack\_varint\_i32(buffer, in\_len)) return -1;

size\_t n = fwrite(in, 1, in\_len, buffer);

if (n != in\_len) return -1;

return 0;

}

int BlockFormatWriter::write\_int(FILE \*buffer, int value) {

uint8\_t buff[4];

buff[0] = value & 0xff;

buff[1] = (value >> 8) & 0xff;

buff[2] = (value >> 16) & 0xff;

buff[3] = (value >> 24) & 0xff;

size\_t n = fwrite(buff, 1, 4, buffer);

if (n != 4) {

return -1;

}

return 0;

}

int BlockFormatWriter::write(const char \*class\_name, const char \*record, int record\_len) {

if (record) {

record\_index\_++;

// The buffer holds max records\_per\_block\_ of records (block).

FILE \*buffer = fopen(temp\_file\_name\_, "a");

if (!buffer) return -1;

if (ftell(buffer) == 0) {

if (pack\_tag\_and\_wiretype(buffer, 1, WIRE\_TYPE\_VARINT))

throw std::invalid\_argument("Error writting tag and wiretype");

if (pack\_varint\_i32(buffer, 1))

throw std::invalid\_argument("Error writting varint\_i32");

if (pack\_tag\_and\_wiretype(buffer, 2, WIRE\_TYPE\_LENGTH\_PREFIXED))

throw std::invalid\_argument("Error writting tag and wiretype");

if (pack\_string(buffer, class\_name, strlen(class\_name)))

throw std::invalid\_argument("Error writting class name");

}

if (pack\_tag\_and\_wiretype(buffer, 3, WIRE\_TYPE\_LENGTH\_PREFIXED))

throw std::invalid\_argument("Error writtig tag and wiretype");

if (pack\_string(buffer, record, record\_len))

throw std::invalid\_argument("Error writting record");

fclose(buffer);

}

if ((record\_index\_ % records\_per\_block\_) == 0) {

flush();

}

return 0;

}

int BlockFormatWriter::flush() {

// Flush the records in the buffer to outputfile

FILE \*buffer = fopen(temp\_file\_name\_, "r");

if (buffer) {

fseek(buffer, 0, SEEK\_END);

int64\_t block\_size = ftell(buffer);

fseek(buffer, 0, SEEK\_SET);

if (fwrite(\_marker, sizeof(\_marker), 1, outputfile\_) != 1) return 1;

if (write\_int(outputfile\_, block\_size)) return 1;

uint8\_t buff[4096];

while (1) {

size\_t n = fread(buff, 1, sizeof(buff), buffer);

if (n) {

size\_t x = fwrite(buff, 1, n, outputfile\_);

if (x != n) return 1;

}

if (n != sizeof(buff)) break;

}

fclose(buffer);

// Remove the buffer

if (remove(temp\_file\_name\_)) return 1;

}

return 0;

}

block\_format\_writer BlockFormatWriter::getHandle() {

return reinterpret\_cast<block\_format\_writer>(this);

}

BlockFormatWriter \*getBlockFormatWriter(block\_format\_writer w) {

return reinterpret\_cast<BlockFormatWriter \*>(w);

}

} // namespace twml

twml\_err block\_format\_writer\_create(block\_format\_writer \*w, const char \*file\_name, int records\_per\_block) {

HANDLE\_EXCEPTIONS(

twml::BlockFormatWriter \*writer = new twml::BlockFormatWriter(file\_name, records\_per\_block);

\*w = reinterpret\_cast<block\_format\_writer>(writer););

return TWML\_ERR\_NONE;

}

twml\_err block\_format\_write(block\_format\_writer w, const char \*class\_name, const char \*record, int record\_len) {

HANDLE\_EXCEPTIONS(

twml::BlockFormatWriter \*writer = twml::getBlockFormatWriter(w);

writer->write(class\_name, record, record\_len););

return TWML\_ERR\_NONE;

}

twml\_err block\_format\_flush(block\_format\_writer w) {

HANDLE\_EXCEPTIONS(

twml::BlockFormatWriter \*writer = twml::getBlockFormatWriter(w);

writer->flush(););

return TWML\_ERR\_NONE;

}

twml\_err block\_format\_writer\_delete(const block\_format\_writer w) {

HANDLE\_EXCEPTIONS(

delete twml::getBlockFormatWriter(w););

return TWML\_ERR\_NONE;

}