#pragma once

#ifdef \_\_cplusplus

#include <twml/defines.h>

#include <cstdint>

#include <cstddef>

#include <cstring>

namespace twml {

class ThriftReader {

protected:

const uint8\_t \*m\_buffer;

public:

ThriftReader(const uint8\_t \*buffer): m\_buffer(buffer) {}

const uint8\_t \*getBuffer() { return m\_buffer; }

void setBuffer(const uint8\_t \*buffer) { m\_buffer = buffer; }

template<typename T> T readDirect() {

T val;

memcpy(&val, m\_buffer, sizeof(T));

m\_buffer += sizeof(T);

return val;

}

template<typename T> void skip() {

m\_buffer += sizeof(T);

}

void skipLength(size\_t length) {

m\_buffer += length;

}

uint8\_t readByte();

int16\_t readInt16();

int32\_t readInt32();

int64\_t readInt64();

double readDouble();

template<typename T> inline

int32\_t getRawBuffer(const uint8\_t \*\*begin) {

int32\_t length = readInt32();

\*begin = m\_buffer;

skipLength(length \* sizeof(T));

return length;

}

};

}

#endif