#define FUSE\_USE\_VERSION 26

#include <fuse.h>

#include <stdio.h>

#include <unistd.h>

#include <sys/types.h>

#include <time.h>

#include <string.h>

#include <stdlib.h>

#include <errno.h>

#include "wad.h"

#include "wad.cpp"

using namespace std;

Wad \*globalWad; *//global*

static int do\_getattr( const char \*path, struct stat \*st )

{

printf( "[getattr] Called\n" );

printf( "\tAttributes of %s requested\n", path );

string temp\_path(path);

if ( globalWad->isDirectory(temp\_path) )

{

st->st\_mode = S\_IFDIR | 0555; *//mode changed from:0755*

st->st\_nlink = 2;

return 0;

}

else if(globalWad->isContent(temp\_path))

{

st->st\_mode = S\_IFREG | 0444; *//mode changed from:0644*

st->st\_nlink = 1;

st->st\_size = globalWad->getSize(temp\_path);

return 0;

}else{

return -errno;

}

}

static int do\_readdir( const char \*path, void \*buffer, fuse\_fill\_dir\_t filler, off\_t offset, struct fuse\_file\_info \*fi )

{

printf( "--> Getting The List of Files of %s\n", path );

filler( buffer, ".", NULL, 0 ); *// Current Directory*

filler( buffer, "..", NULL, 0 ); *// Parent Directory*

string temp\_path(path);

if(globalWad->isDirectory(temp\_path)){

vector<string> entries;

globalWad->getDirectory(temp\_path, &entries);

cout << "Make sure that I got inside" << endl;

for(int i = 0; i < static\_cast<int>(entries.size()); i++){

filler(buffer, entries.at(i).c\_str(), NULL, 0);

}

cout << "Make sure that I got here (readdir)" << endl;

return 0;

}else{

return -1;

}

}

static int do\_read( const char \*path, char \*buffer, size\_t size, off\_t offset, struct fuse\_file\_info \*fi )

{

string temp\_path(path);

*//int mysize = 0;*

*//char \*selectedText = NULL;*

if ( globalWad->isContent(temp\_path) ){

int mysize = globalWad->getContents(temp\_path, buffer, globalWad->getSize(temp\_path), 0);

return mysize;

}

else{

return -errno;

}

}

static int do\_open(const char \*path, struct fuse\_file\_info \*fi)

{

string temp\_path(path);

if(globalWad->isContent(temp\_path)){

return 0;

}else{

return -errno;

}

}

static int do\_opendir(const char \*path, struct fuse\_file\_info \*fi)

{

string temp\_path(path);

if(globalWad->isDirectory(temp\_path)){

return 0;

}else{

return -errno;

}

return 0;

}

static int do\_release(const char \*path, struct fuse\_file\_info \*fi)

{

string temp\_path(path);

if(globalWad->isContent(temp\_path)){

return 0;

}else{

return -errno;

}

}

static int do\_releasedir(const char \*path, struct fuse\_file\_info \*fi)

{

string temp\_path(path);

if(globalWad->isDirectory(temp\_path)){

return 0;

}else{

return -errno;

}

}

struct fuse\_operations operations; *//global fuse\_operations*

int main( int argc, char \*argv[] )

{

if (argc < 2)

{

cout << "No file specified. Exiting." << endl;

exit(EXIT\_SUCCESS);

}

globalWad = Wad::loadWad(argv[1]);

*//fuse\_operations operations;*

operations.getattr = do\_getattr;

operations.readdir = do\_readdir;

operations.read = do\_read;

operations.open = do\_open;

operations.opendir = do\_opendir;

operations.release = do\_release;

operations.releasedir = do\_releasedir;

*//char\* debugFlag[] = {"-d"};*

*//char \*temp\_argv[3] = {argv[0], argv[2], argv[3]};*

char \*temp\_argv[2] = {argv[0], argv[2]};

int arrayLength = 2;

return fuse\_main( arrayLength, temp\_argv, &operations, NULL );

*//return fuse\_main( arrayLength, temp\_argv, &operations, globalWad );*

*// int fuseReturn = fuse\_main( arrayLength, temp\_argv, operations, NULL );*

*// delete operations;*

*// delete globalWad;*

*// return fuseReturn;*

}