#include <iostream>

#include "windows.h"

#include <string>

#include "ClientMS.h"

using namespace std;

string GetErrorMail(int code)

{

string msgText = "";

switch (code)

{

case WSAEINTR: msgText = "WSAEINTR"; break;

case WSAEACCES: msgText = "WSAEACCES"; break;

case WSAEFAULT: msgText = "WSAEFAULT"; break;

default: msgText = "Error";

};

return msgText;

} определение функции, которая принимает код ошибки и возвращает строку с текстом ошибки.

string SetErrorMail(string msgText, int code)

{

return msgText + GetErrorMail(code);

}

определение функции, которая принимает строку с текстом ошибки и код ошибки, и возвращает строку с текстом ошибки, включающую переданный текст и код ошибки.

int main()

{

setlocale(LC\_ALL, "rus");

try {

HANDLE clientMailSlot; объявление переменной типа HANDLE для хранения дескриптора клиентского mailslot.

double t1, t2;

if ((clientMailSlot = CreateFile(L"\\\\\*\\mailslot\\BOX",

GENERIC\_WRITE,

FILE\_SHARE\_READ | FILE\_SHARE\_WRITE,

NULL,

OPEN\_EXISTING,

NULL,

NULL)) == INVALID\_HANDLE\_VALUE)

throw SetErrorMail("CreateFile: ", GetLastError());

создание дескриптора клиентского mailslot с указанными параметрами и сохранение его в переменную clientMailSlot. Если функция CreateFile вернула INVALID\_HANDLE\_VALUE, выбрасывается исключение с сообщением об ошибке, полученном с помощью GetLastError.

Функция CreateFile используется для создания или открытия файлов, каталогов, дисков и других объектов файловой системы. В данном коде функция используется для открытия существующего именованного канала.

Параметры функции:

L"\\\\\*\\mailslot\\BOX" - имя именованного канала, в данном случае это строка символов в формате LPTSTR, которая задает полное имя канала.

GENERIC\_WRITE - режим доступа, который устанавливает права на запись в файл. В данном случае, файл можно записать.

FILE\_SHARE\_READ | FILE\_SHARE\_WRITE - флаг, который указывает, какие другие операции могут быть совместно использованы с файлом, открытым этой функцией. В данном случае, устанавливаются права на чтение и запись файла совместно с другими приложениями.

NULL - указатель на атрибуты безопасности (SECURITY\_ATTRIBUTES), которые будут использоваться для установки атрибутов защиты на открытый файл. В данном случае, эти атрибуты не заданы.

OPEN\_EXISTING - флаг, который определяет, что файл, на который указывает имя, уже существует. В данном случае, канал уже создан и открыт на серверной стороне.

NULL - указатель на шаблон файлового объекта. В данном случае, шаблон не используется.

NULL - дескриптор файла. В данном случае, он не используется.

В итоге функция создает дескриптор файла и возвращает его значение. Если файл не может быть открыт или создан, функция возвращает INVALID\_HANDLE\_VALUE.

cout << "Hello i am Client" << endl;

char writeBuf[50] = "Hello from Client-Mailslot";

DWORD writeMsg;

t1 = clock();

for (int i = 1; i <= 30; i++) {

if(!WriteFile(clientMailSlot,writeBuf,sizeof(writeBuf),&writeMsg,NULL))

throw SetErrorMail("WriteFile: ", GetLastError());

//Sleep(10);

cout << "Message " << i << " was sent" << endl;

}WriteFile - это функция WinAPI для записи данных в файл, в том числе и для записи в mailslot, как в данном случае.

Параметры функции:

clientMailSlot - дескриптор файла, который был создан ранее и представляет собой mailslot, в который будет производиться запись.

writeBuf - указатель на буфер с данными, которые будут записаны в mailslot.

sizeof(writeBuf) - количество байт, которые нужно записать в mailslot из буфера.

&writeMsg - указатель на переменную, в которую будет сохранено количество записанных байтов.

NULL - указатель на структуру OVERLAPPED, которая используется для асинхронных операций ввода/вывода. В данном случае, операция ввода/вывода выполняется в синхронном режиме, поэтому указатель на OVERLAPPED передаётся как NULL.

Функция возвращает значение TRUE при успешной записи в mailslot и FALSE в случае ошибки, которая может быть получена с помощью функции GetLastError().

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

if (!WriteFile(clientMailSlot, writeBuf, sizeof(writeBuf), &writeMsg, NULL)) {

}

}

t2 = clock();

Sleep(3000);

if (!CloseHandle(clientMailSlot))

throw "Error: CloseHandle";

cout << endl << "Время передачи: " << (t2 - t1) / 1000 << " сек." << endl << endl;

system("pause");

}

catch (string e) {

cout << e << endl;

}

return 0;

}