ИМЕННОВАННЫЕ КАНАЛЫ (PIPES)

Канал – это механизм передачи сообщений от одного процесса другому (независимому от первого процесса). Таким образом, канал реализует механизм межпроцессного взаимодействия.

Серверная часть:

#include"stdafx.h"

#include<Windows.h>

#include<string>

#include <iostream>

int \_tmain(int argc, \_TCHAR\* argv[])

{

HANDLE hPipe;

char buffer[1024] = "";

DWORD dwRead;

int retval = 0;

hPipe = CreateNamedPipe(TEXT("\\\\.\\pipe\\Pipe"),

PIPE\_ACCESS\_DUPLEX/\* Для чтения и записи\*/ | PIPE\_TYPE\_BYTE /\*передача байт \*/| PIPE\_READMODE\_BYTE,/\*Канал открывается в режиме последовательной передачи отдельных байт\*/

PIPE\_WAIT, 1/\*кол-во каналов\*/, 1024 \* 16, 1024 \* 16,

NMPWAIT\_USE\_DEFAULT\_WAIT, NULL);

while (hPipe != INVALID\_HANDLE\_VALUE)

{

if (ConnectNamedPipe(hPipe, NULL) != FALSE)

{

while (1) {

retval = ReadFile(hPipe, buffer, sizeof(buffer) - 1, &dwRead, NULL);

if(retval)

{

buffer[dwRead] = '\0';

std::cout << "Client: " << buffer << std::endl;

}

else

{

std::cout << "Unable to write\n";

return 1;

}

std::string userBuff = "";

std::cout << "Enter text: ";

std::getline(std::cin, userBuff);

retval = WriteFile(hPipe, userBuff.c\_str(), userBuff.length(), &dwRead, NULL);

if (!retval)

{

std::cout << "Unable to write\n";

return 1;

}

}

}

DisconnectNamedPipe(hPipe);

}

return 0;

}

Клиентская часть:

#include "stdafx.h"

#include <iostream>

#include <string>

#include <Windows.h>

int \_tmain(int argc, \_TCHAR\* argv[])

{

HANDLE hPipe;

DWORD dwRead;

DWORD dwWritten;

char buffer[1024];

int retval = 0;

hPipe = CreateFile(TEXT("\\\\.\\pipe\\Pipe"), GENERIC\_READ | GENERIC\_WRITE,0,NULL,OPEN\_EXISTING,0,NULL);

if (hPipe != INVALID\_HANDLE\_VALUE)

{

while (1) {

std::string userBuff = "";

std::cout << "Enter text: ";

std::getline(std::cin, userBuff);

retval = WriteFile(hPipe, userBuff.c\_str(), userBuff.length(), &dwRead, NULL);

if (!retval)

{

std::cout << "Unable to write\n";

return 1;

}

retval = ReadFile(hPipe, buffer, sizeof(buffer) - 1, &dwRead, NULL);

if (retval)

{

buffer[dwRead] = '\0';

std::cout << "Server: " << buffer << std::endl;

}

else

{

std::cout << "Unable to read\n";

return 1;

}

}

CloseHandle(hPipe);

}

return 0;

}