[Lab06]대화상자기반기

DialogBase2Dlg.cpp

BOOL CDialogBase2Dlg::OnInitDialog()

{

CDialogEx::OnInitDialog();

// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

// IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

BOOL bNameValid;

CString strAboutMenu;

bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);

ASSERT(bNameValid);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는

// 프레임워크가 이 작업을 자동으로 수행합니다.

SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.

SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.

// TODO: 여기에 추가 초기화 작업을 추가합니다.

m\_combo.AddString(\_T("+"));

m\_combo.AddString(\_T("-"));

m\_combo.AddString(\_T("\*"));

m\_combo.AddString(\_T("/"));

return TRUE; // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.

}

void CDialogBase2Dlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialogEx::OnSysCommand(nID, lParam);

}

}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면

// 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는

// 프레임워크에서 이 작업을 자동으로 수행합니다.

void CDialogBase2Dlg::OnPaint()

{

if (IsIconic())

{

CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);

// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.

int cxIcon = GetSystemMetrics(SM\_CXICON);

int cyIcon = GetSystemMetrics(SM\_CYICON);

CRect rect;

GetClientRect(&rect);

int x = (rect.Width() - cxIcon + 1) / 2;

int y = (rect.Height() - cyIcon + 1) / 2;

// 아이콘을 그립니다.

dc.DrawIcon(x, y, m\_hIcon);

}

else

{

CDialogEx::OnPaint();

}

}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서

// 이 함수를 호출합니다.

HCURSOR CDialogBase2Dlg::OnQueryDragIcon()

{

return static\_cast<HCURSOR>(m\_hIcon);

}

void CDialogBase2Dlg::OnBnClickedStart() //사용자가"계산"버튼 눌렀을때

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

CString CStr\_x1, CStr\_x2, Cstr, print\_y;

double x1, x2, y;

GetDlgItem(IDC\_EDIT1)->GetWindowText(CStr\_x1); //IDC\_EDIT1에 입력된 값 읽어와서 CStr\_x1에 저장

x1 = \_ttof(CStr\_x1); //CString을 배정도형으로 변환하여 x1에 저장

GetDlgItem(IDC\_EDIT2)->GetWindowText(CStr\_x2); //IDC\_EDIT2의 입력된 값 읽어와서 CStr\_x2에 저장

x2 = \_ttof(CStr\_x2); //CString을 배정도형으로 변환하여 x2에 저장

m\_combo.GetLBText(m\_combo.GetCurSel(), Cstr); //콤보박스에서 값읽어 와서 Cstr에 저장

if (Cstr == '+') //덧셈

{

y = x1 + x2;

print\_y.Format(\_T("%.2lf"), y); //자료형변경

m\_edit3.SetWindowTextW(print\_y); //3번째 edit창에 출력

}

else if (Cstr == '-') //뺄셈

{

y = x1 - x2;

print\_y.Format(\_T("%.2lf"), y);

m\_edit3.SetWindowTextW(print\_y); //3번째 edit창에 출력

}

else if (Cstr == '\*') //곱셈

{

y = x1 \* x2;

print\_y.Format(\_T("%.2lf"), y);

m\_edit3.SetWindowTextW(print\_y); //3번째 edit창에 출력

}

else if (Cstr == '/') //나눗셈

{

y = x1 / x2;

if (x2 == 0) //0으로 나누려고하면

{

MessageBox(\_T("잘못된 입력입니다"), \_T("ERROR"), MB\_ICONERROR); //오류 메시지 띄우기

return;

}

print\_y.Format(\_T("%.2lf"), y); //자료형변경

m\_edit3.SetWindowTextW(print\_y); //3번째 edit창에 출력

}

}

void CDialogBase2Dlg::OnBnClickedClear() //"CLEAR"버튼 눌렀을때

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

//클리어버튼 눌리면 edit창의 text를 모두 0으로

m\_edit1.SetWindowTextW(\_T("0"));

m\_edit2.SetWindowTextW(\_T("0"));

m\_edit3.SetWindowTextW(\_T("0"));

}

[Lab07] Timer Circle

1. WorkerThread2View.cpp

// WorkerThread2View.cpp : CWorkerThread2View 클래스의 구현

//

#include "stdafx.h"

// SHARED\_HANDLERS는 미리 보기, 축소판 그림 및 검색 필터 처리기를 구현하는 ATL 프로젝트에서 정의할 수 있으며

// 해당 프로젝트와 문서 코드를 공유하도록 해 줍니다.

#ifndef SHARED\_HANDLERS

#include "WorkerThread2.h"

#endif

#include "WorkerThread2Doc.h"

#include "WorkerThread2View.h"

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

// CWorkerThread2View

IMPLEMENT\_DYNCREATE(CWorkerThread2View, CView)

BEGIN\_MESSAGE\_MAP(CWorkerThread2View, CView)

// 표준 인쇄 명령입니다.

ON\_COMMAND(ID\_FILE\_PRINT, &CView::OnFilePrint)

ON\_COMMAND(ID\_FILE\_PRINT\_DIRECT, &CView::OnFilePrint)

ON\_COMMAND(ID\_FILE\_PRINT\_PREVIEW, &CView::OnFilePrintPreview)

ON\_WM\_LBUTTONDOWN()

END\_MESSAGE\_MAP()

// CWorkerThread2View 생성/소멸

CWorkerThread2View::CWorkerThread2View()

{

// TODO: 여기에 생성 코드를 추가합니다.

}

CWorkerThread2View::~CWorkerThread2View()

{

}

BOOL CWorkerThread2View::PreCreateWindow(CREATESTRUCT& cs)

{

// TODO: CREATESTRUCT cs를 수정하여 여기에서

// Window 클래스 또는 스타일을 수정합니다.

return CView::PreCreateWindow(cs);

}

// CWorkerThread2View 그리기

void CWorkerThread2View::OnDraw(CDC\* /\*pDC\*/)

{

CWorkerThread2Doc\* pDoc = GetDocument();

ASSERT\_VALID(pDoc);

if (!pDoc)

return;

// TODO: 여기에 원시 데이터에 대한 그리기 코드를 추가합니다.

}

// CWorkerThread2View 인쇄

BOOL CWorkerThread2View::OnPreparePrinting(CPrintInfo\* pInfo)

{

// 기본적인 준비

return DoPreparePrinting(pInfo);

}

void CWorkerThread2View::OnBeginPrinting(CDC\* /\*pDC\*/, CPrintInfo\* /\*pInfo\*/)

{

// TODO: 인쇄하기 전에 추가 초기화 작업을 추가합니다.

}

void CWorkerThread2View::OnEndPrinting(CDC\* /\*pDC\*/, CPrintInfo\* /\*pInfo\*/)

{

// TODO: 인쇄 후 정리 작업을 추가합니다.

}

// CWorkerThread2View 진단

#ifdef \_DEBUG

void CWorkerThread2View::AssertValid() const

{

CView::AssertValid();

}

void CWorkerThread2View::Dump(CDumpContext& dc) const

{

CView::Dump(dc);

}

CWorkerThread2Doc\* CWorkerThread2View::GetDocument() const // 디버그되지 않은 버전은 인라인으로 지정됩니다.

{

ASSERT(m\_pDocument->IsKindOf(RUNTIME\_CLASS(CWorkerThread2Doc)));

return (CWorkerThread2Doc\*)m\_pDocument;

}

#endif //\_DEBUG

#define LOOPCNT (200000)

#define DIVCNT (LOOPCNT/500)

UINT MyDraw(LPVOID arg)

{

CirTimerThread \*pArg = (CirTimerThread\*)arg;

CClientDC dc(pArg->pWnd);

CPen WHITEpen(PS\_SOLID, 2, RGB(255, 255, 255)); //흰색

for (int i = 0; i < LOOPCNT; i++) //윈도우 크기가 바뀔때마다 다시 그릴 수 있도록 for-loop으로

{

dc.Ellipse(pArg->pos.x - 20, pArg->pos.y - 20, pArg->pos.x + 20, pArg->pos.y + 20 ); //반지름 2인 원

}

Sleep(3000); //3000ms => 3sec 기다리도록

dc.SelectObject(&WHITEpen);

dc.Ellipse(pArg->pos.x - 20, pArg->pos.y - 20, pArg->pos.x + 20, pArg->pos.y + 20); //그렸던 원 위치에 흰색 원 그려서 원 지우기

pArg->EndThread(); /////스레드 종료

return 0;

} //그렸던 원이 지워지고난 후 스레드 종료E

// CWorkerThread2View 메시지 처리기

void CWorkerThread2View::OnLButtonDown(UINT nFlags, CPoint point)

{

CView::OnLButtonDown(nFlags, point);

//화면지우기

CClientDC dc(this);

dc.SelectStockObject(WHITE\_PEN);

dc.SelectStockObject(WHITE\_BRUSH);

//왼쪽마우스버튼 누르면 스레드 생성

CirTimerThread \*arg = new CirTimerThread(this, point); //동적 스레드 생성

arg->GenerateThread();

}

void CirTimerThread::GenerateThread()

{

AfxBeginThread(MyDraw, this); //스레드 동적 생성. CWinThread타입 스레드 객체 주소값 반환

}

void CirTimerThread::EndThread()

{

AfxEndThread(0, TRUE); //스레드 강제 종료 함수

}

1. WorkerThread2.cpp
2. // WorkerThread2.cpp : 응용 프로그램에 대한 클래스 동작을 정의합니다.
3. //
4. #include "stdafx.h"
5. #include "afxwinappex.h"
6. #include "afxdialogex.h"
7. #include "WorkerThread2.h"
8. #include "MainFrm.h"
9. #include "WorkerThread2Doc.h"
10. #include "WorkerThread2View.h"
11. #ifdef \_DEBUG
12. #define new DEBUG\_NEW
13. #endif
14. // CWorkerThread2App
15. BEGIN\_MESSAGE\_MAP(CWorkerThread2App, CWinApp)
16. ON\_COMMAND(ID\_APP\_ABOUT, &CWorkerThread2App::OnAppAbout)
17. // 표준 파일을 기초로 하는 문서 명령입니다.
18. ON\_COMMAND(ID\_FILE\_NEW, &CWinApp::OnFileNew)
19. ON\_COMMAND(ID\_FILE\_OPEN, &CWinApp::OnFileOpen)
20. // 표준 인쇄 설정 명령입니다.
21. ON\_COMMAND(ID\_FILE\_PRINT\_SETUP, &CWinApp::OnFilePrintSetup)
22. END\_MESSAGE\_MAP()
23. // CWorkerThread2App 생성
24. CWorkerThread2App::CWorkerThread2App()
25. {
26. // TODO: 아래 응용 프로그램 ID 문자열을 고유 ID 문자열로 바꾸십시오(권장).
27. // 문자열에 대한 서식: CompanyName.ProductName.SubProduct.VersionInformation
28. SetAppID(\_T("WorkerThread2.AppID.NoVersion"));
29. // TODO: 여기에 생성 코드를 추가합니다.
30. // InitInstance에 모든 중요한 초기화 작업을 배치합니다.
31. }
32. UINT CPUBusy(LPVOID arg)
33. {
34. while (1);
35. return 0;
36. }
37. // 유일한 CWorkerThread2App 개체입니다.
38. CWorkerThread2App theApp;
39. // CWorkerThread2App 초기화
40. BOOL CWorkerThread2App::InitInstance()
41. {
42. // 응용 프로그램 매니페스트가 ComCtl32.dll 버전 6 이상을 사용하여 비주얼 스타일을
43. // 사용하도록 지정하는 경우, Windows XP 상에서 반드시 InitCommonControlsEx()가 필요합니다.
44. // InitCommonControlsEx()를 사용하지 않으면 창을 만들 수 없습니다.
45. INITCOMMONCONTROLSEX InitCtrls;
46. InitCtrls.dwSize = sizeof(InitCtrls);
47. // 응용 프로그램에서 사용할 모든 공용 컨트롤 클래스를 포함하도록
48. // 이 항목을 설정하십시오.
49. InitCtrls.dwICC = ICC\_WIN95\_CLASSES;
50. InitCommonControlsEx(&InitCtrls);
51. CWinApp::InitInstance();
52. EnableTaskbarInteraction(FALSE);
53. // RichEdit 컨트롤을 사용하려면 AfxInitRichEdit2()가 있어야 합니다.
54. // AfxInitRichEdit2();
55. // 표준 초기화
56. // 이들 기능을 사용하지 않고 최종 실행 파일의 크기를 줄이려면
57. // 아래에서 필요 없는 특정 초기화
58. // 루틴을 제거해야 합니다.
59. // 해당 설정이 저장된 레지스트리 키를 변경하십시오.
60. // TODO: 이 문자열을 회사 또는 조직의 이름과 같은
61. // 적절한 내용으로 수정해야 합니다.
62. SetRegistryKey(\_T("로컬 응용 프로그램 마법사에서 생성된 응용 프로그램"));
63. LoadStdProfileSettings(4); // MRU를 포함하여 표준 INI 파일 옵션을 로드합니다.
64. // 응용 프로그램의 문서 템플릿을 등록합니다. 문서 템플릿은
65. // 문서, 프레임 창 및 뷰 사이의 연결 역할을 합니다.
66. CSingleDocTemplate\* pDocTemplate;
67. pDocTemplate = new CSingleDocTemplate(
68. IDR\_MAINFRAME,
69. RUNTIME\_CLASS(CWorkerThread2Doc),
70. RUNTIME\_CLASS(CMainFrame), // 주 SDI 프레임 창입니다.
71. RUNTIME\_CLASS(CWorkerThread2View));
72. if (!pDocTemplate)
73. return FALSE;
74. AddDocTemplate(pDocTemplate);
75. // 표준 셸 명령, DDE, 파일 열기에 대한 명령줄을 구문 분석합니다.
76. CCommandLineInfo cmdInfo;
77. ParseCommandLine(cmdInfo);
78. // 명령줄에 지정된 명령을 디스패치합니다.
79. // 응용 프로그램이 /RegServer, /Register, /Unregserver 또는 /Unregister로 시작된 경우 FALSE를 반환합니다.
80. if (!ProcessShellCommand(cmdInfo))
81. return FALSE;
82. // 창 하나만 초기화되었으므로 이를 표시하고 업데이트합니다.
83. m\_pMainWnd->ShowWindow(SW\_SHOW);
84. m\_pMainWnd->UpdateWindow();
85. SYSTEM\_INFO si;
86. GetSystemInfo(&si);
87. for (int i = 0; i < (int)si.dwNumberOfProcessors - 1; i++)
88. {
89. AfxBeginThread(CPUBusy, NULL, THREAD\_PRIORITY\_HIGHEST, 0, 0, NULL);;
90. }
91. return TRUE;
92. }
93. // CWorkerThread2App 메시지 처리기
94. // 응용 프로그램 정보에 사용되는 CAboutDlg 대화 상자입니다.
95. class CAboutDlg : public CDialogEx
96. {
97. public:
98. CAboutDlg();
99. // 대화 상자 데이터입니다.
100. #ifdef AFX\_DESIGN\_TIME
101. enum { IDD = IDD\_ABOUTBOX };
102. #endif
103. protected:
104. virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 지원입니다.
105. // 구현입니다.
106. protected:
107. DECLARE\_MESSAGE\_MAP()
108. };
109. CAboutDlg::CAboutDlg() : CDialogEx(IDD\_ABOUTBOX)
110. {
111. }
112. void CAboutDlg::DoDataExchange(CDataExchange\* pDX)
113. {
114. CDialogEx::DoDataExchange(pDX);
115. }
116. BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)
117. END\_MESSAGE\_MAP()
118. // 대화 상자를 실행하기 위한 응용 프로그램 명령입니다.
119. void CWorkerThread2App::OnAppAbout()
120. {
121. CAboutDlg aboutDlg;
122. aboutDlg.DoModal();
123. }
124. // CWorkerThread2App 메시지 처리기

[Lab08] TCP 기반 양방향 통신

1. TCPServer\_thdDlg.cpp
2. UINT RXThread(LPVOID arg)
3. {
4. ThreadArg \*pArg = (ThreadArg\*)arg;
5. CStringList \*plist = pArg->pList;
6. CTCPServer\_thdDlg \*pDlg = (CTCPServer\_thdDlg \*)pArg->pDlg;
7. while (pArg->Thread\_run)
8. {
9. POSITION pos = plist->GetHeadPosition();
10. POSITION current\_pos;
11. while (pos != NULL)
12. {
13. current\_pos = pos;
14. rx\_cs.Lock();
15. CString str = plist->GetNext(pos);
16. rx\_cs.Unlock();
17. CString message;
18. pDlg->m\_rx\_edit.GetWindowText(message);
19. message += str;
20. pDlg->m\_rx\_edit.SetWindowTextW(message);
21. pDlg->m\_rx\_edit.LineScroll(pDlg->m\_rx\_edit.GetLineCount());
22. plist->RemoveAt(current\_pos);
23. }
24. Sleep(10);
25. }
26. return 0;
27. }
28. UINT TXThread(LPVOID arg)
29. {
30. //////////////
31. ThreadArg \*pArg = (ThreadArg \*)arg; // ThreadArg 객체 생성
32. CStringList \*plist = pArg->pList;
33. CTCPServer\_thdDlg \*pDlg = (CTCPServer\_thdDlg\*)pArg->pDlg;
34. while (pArg->Thread\_run) //스레드가 run상태일때만 동작
35. {
36. POSITION pos = plist->GetHeadPosition();
37. POSITION current\_pos;
38. while (pos != NULL) {
39. current\_pos = pos;
41. rx\_cs.Lock();
42. CString str = plist->GetNext(pos);
43. rx\_cs.Unlock();
45. CString message;
46. pDlg->m\_tx\_edit.GetWindowText(message); //입력받은 메세지를 얻어서
47. //message += "\r\n";
48. pDlg->m\_tx\_edit.SetWindowTextW(message); // 메세지를 출력
49. pDlg->m\_pDataSocket->Send(str, (str.GetLength() + 1) \* sizeof(TCHAR));
50. pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount());
51. plist->RemoveAt(current\_pos); // 현재위치의 plist 삭제
52. }
53. Sleep(10);
54. }
55. return 0;
56. }
57. BOOL CTCPServer\_thdDlg::OnInitDialog()
58. {
59. CDialogEx::OnInitDialog();
60. // 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.
61. // IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.
62. ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);
63. ASSERT(IDM\_ABOUTBOX < 0xF000);
64. CMenu\* pSysMenu = GetSystemMenu(FALSE);
65. if (pSysMenu != NULL)
66. {
67. BOOL bNameValid;
68. CString strAboutMenu;
69. bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);
70. ASSERT(bNameValid);
71. if (!strAboutMenu.IsEmpty())
72. {
73. pSysMenu->AppendMenu(MF\_SEPARATOR);
74. pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);
75. }
76. }
77. // 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는
78. // 프레임워크가 이 작업을 자동으로 수행합니다.
79. SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.
80. SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.
81. // TODO: 여기에 추가 초기화 작업을 추가합니다.
82. //TX queue
83. CStringList \*newlist1 = new CStringList;
84. arg1.pList = newlist1;
85. arg1.Thread\_run = 1; //스레드를 실행 상태로 표시
86. arg1.pDlg = this;
87. //RX queue
88. CStringList\* newlist2 = new CStringList;
89. arg2.pList = newlist2;
90. arg2.Thread\_run = 1; //스레드를 실행 상태로 표시
91. arg2.pDlg = this;
92. WSADATA wsa;
93. int error\_code;
94. if ((error\_code = WSAStartup(MAKEWORD(2, 2), &wsa)) != 0)
95. {
96. TCHAR buffer[256];
97. FormatMessageW(FORMAT\_MESSAGE\_FROM\_SYSTEM, NULL, error\_code,
98. MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT), buffer, 256, NULL);
99. AfxMessageBox(buffer, MB\_ICONERROR);
100. }
101. m\_pListenSocket = NULL;
102. m\_pDataSocket = NULL;
103. ASSERT(m\_pListenSocket == NULL);
104. m\_pListenSocket = new CListenSocket(this);
105. if (m\_pListenSocket->Create(8000))
106. {
107. if (m\_pListenSocket->Listen())
108. {
109. AfxMessageBox(\_T("서버를 시작합니다."), MB\_ICONINFORMATION);
110. pThread1 = AfxBeginThread(TXThread, (LPVOID)&arg1);
111. pThread2 = AfxBeginThread(RXThread, (LPVOID)&arg2);
113. return TRUE;
114. }
115. }
116. else
117. {
118. int err = m\_pListenSocket->GetLastError();
119. TCHAR buffer[256];
120. FormatMessageW(FORMAT\_MESSAGE\_FROM\_SYSTEM, NULL, err,
121. MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT), buffer, 256, NULL);
122. AfxMessageBox(buffer, MB\_ICONERROR);
123. }
124. AfxMessageBox(\_T("이미 실행 중인 서버가 있습니다.")\_T("\n프로그램을 종료합니다."), MB\_ICONERROR);
125. return FALSE; // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.
126. }
127. void CTCPServer\_thdDlg::OnSysCommand(UINT nID, LPARAM lParam)
128. {
129. if ((nID & 0xFFF0) == IDM\_ABOUTBOX)
130. {
131. CAboutDlg dlgAbout;
132. dlgAbout.DoModal();
133. }
134. else
135. {
136. CDialogEx::OnSysCommand(nID, lParam);
137. }
138. }
139. // 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면
140. // 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는
141. // 프레임워크에서 이 작업을 자동으로 수행합니다.
142. void CTCPServer\_thdDlg::OnPaint()
143. {
144. if (IsIconic())
145. {
146. CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.
147. SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);
148. // 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.
149. int cxIcon = GetSystemMetrics(SM\_CXICON);
150. int cyIcon = GetSystemMetrics(SM\_CYICON);
151. CRect rect;
152. GetClientRect(&rect);
153. int x = (rect.Width() - cxIcon + 1) / 2;
154. int y = (rect.Height() - cyIcon + 1) / 2;
155. // 아이콘을 그립니다.
156. dc.DrawIcon(x, y, m\_hIcon);
157. }
158. else
159. {
160. CDialogEx::OnPaint();
161. }
162. }
163. // 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서
164. // 이 함수를 호출합니다.
165. HCURSOR CTCPServer\_thdDlg::OnQueryDragIcon()
166. {
167. return static\_cast<HCURSOR>(m\_hIcon);
168. }
169. void CTCPServer\_thdDlg::ProcessAccept(int nErrorCode)
170. {
171. CString PeerAddr;
172. UINT PeerPort;
173. CString str;
174. ASSERT(nErrorCode == 0);
175. if (m\_pDataSocket == NULL)
176. {
177. m\_pDataSocket = new CDataSocket(this);
178. if (m\_pListenSocket->Accept(\*m\_pDataSocket))
179. {
180. m\_pDataSocket->GetPeerName(PeerAddr, PeerPort);
181. str.Format(\_T("### IP주소: %s, 포트 번호: %d ###\r\n"), PeerAddr, PeerPort);
182. m\_rx\_edit.SetWindowTextW(str);
183. }
184. else
185. {
186. delete m\_pDataSocket;
187. m\_pDataSocket = NULL;
188. }
189. }
190. }
191. void CTCPServer\_thdDlg::ProcessReceive(CDataSocket \*pSocket, int nErrorCode)
192. {
193. TCHAR pBuf[1024 + 1];
194. CString strData;
195. int nbytes;
196. nbytes = pSocket->Receive(pBuf, 1024);
197. pBuf[nbytes] = NULL;
198. strData = (LPCTSTR)pBuf;
199. rx\_cs.Lock();
200. arg2.pList->AddTail((LPCTSTR)strData);
201. rx\_cs.Unlock();
202. }
203. void CTCPServer\_thdDlg::ProcessClose(CDataSocket \*pSocket, int nErrorCode)
204. {
205. pSocket->Close();
206. delete m\_pDataSocket;
207. m\_pDataSocket = NULL;
208. int len = m\_rx\_edit.GetWindowTextLengthW();
209. CString message = \_T("### 접속 종료 ###\r\n");
210. m\_rx\_edit.SetSel(len, len);
211. m\_rx\_edit.ReplaceSel(message);
212. }
213. void CTCPServer\_thdDlg::OnBnClickedSend()
214. {
215. // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
216. CString tx\_message;
217. m\_tx\_edit\_short.GetWindowText(tx\_message); //텍스트 받고
218. tx\_message += \_T("\r\n"); //줄바꿈 위해
219. tx\_cs.Lock();
220. arg1.pList->AddTail(tx\_message);//메시지 담고
221. tx\_cs.Unlock();
222. m\_tx\_edit\_short.SetWindowTextW(\_T(""));
223. m\_tx\_edit\_short.SetFocus();
224. int len = m\_tx\_edit.GetWindowTextLengthW();
225. m\_tx\_edit.SetSel(len, len);
226. m\_tx\_edit.ReplaceSel(tx\_message); //받은 메시지 큰 edit창에 출력
227. }
228. void CTCPServer\_thdDlg::OnBnClickedDisconnect()
229. {
230. // TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.
231. if (m\_pDataSocket == NULL)
232. {
233. AfxMessageBox(\_T("이미 접속 종료"));
234. }
235. else
236. {
237. arg1.Thread\_run = 0;
238. arg2.Thread\_run = 0;
239. m\_pDataSocket->Close();
240. delete m\_pDataSocket;
241. m\_pDataSocket = NULL;
242. m\_pListenSocket->Close();
243. delete m\_pListenSocket;
244. m\_pListenSocket = NULL;
245. }
246. }

2. ListenSocket.cpp

// ListenSocket.cpp : 구현 파일입니다.

//

#include "stdafx.h"

#include "TCPServer\_thd.h"

#include "ListenSocket.h"

#include "TCPServer\_thdDlg.h"

// CListenSocket

CListenSocket::CListenSocket(CTCPServer\_thdDlg \*pDlg)

{

m\_pDlg = pDlg;

}

CListenSocket::~CListenSocket()

{

}

// CListenSocket 멤버 함수

void CListenSocket::OnAccept(int nErrorCode)

{

// TODO: 여기에 특수화된 코드를 추가 및/또는 기본 클래스를 호출합니다.

CSocket::OnAccept(nErrorCode);

m\_pDlg->ProcessAccept(nErrorCode);

}

. DataSocket.cpp

// DataSocket.cpp : 구현 파일입니다.

//

#include "stdafx.h"

#include "TCPServer\_thd.h"

#include "DataSocket.h"

#include "TCPServer\_thdDlg.h"

// CDataSocket

CDataSocket::CDataSocket(CTCPServer\_thdDlg \*pDlg)

{

m\_pDlg = pDlg;

}

CDataSocket::~CDataSocket()

{

}

// CDataSocket 멤버 함수

void CDataSocket::OnReceive(int nErrorCode)

{

CSocket::OnReceive(nErrorCode);

m\_pDlg->ProcessReceive(this, nErrorCode);

}

void CDataSocket::OnClose(int nErrorCode)

{

CSocket::OnClose(nErrorCode);

m\_pDlg->ProcessClose(this, nErrorCode);

}

TCPClient\_thdDlg.cpp

UINT RXThread(LPVOID arg)

{

ThreadArg \*pArg = (ThreadArg\*)arg;

CStringList \*plist = pArg->pList;

CTCPClient\_thdDlg \*pDlg = (CTCPClient\_thdDlg \*)pArg->pDlg;

while (pArg->Thread\_run)

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

current\_pos = pos;

rx\_cs.Lock();

CString str = plist->GetNext(pos);

rx\_cs.Unlock();

CString message;

pDlg->m\_rx\_edit.GetWindowText(message);

message += str;

pDlg->m\_rx\_edit.SetWindowTextW(message);

pDlg->m\_rx\_edit.LineScroll(pDlg->m\_rx\_edit.GetLineCount());

plist->RemoveAt(current\_pos);

}

Sleep(10);

}

return 0;

}

UINT TXThread(LPVOID arg)

{

ThreadArg \*pArg = (ThreadArg \*)arg;

CStringList \*plist = pArg->pList;

CTCPClient\_thdDlg \*pDlg = (CTCPClient\_thdDlg\*)pArg->pDlg;

while (pArg->Thread\_run) // 스레드가 run상태일때만 동작

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL) {

current\_pos = pos;

tx\_cs.Lock();

CString str = plist->GetNext(pos);

tx\_cs.Unlock();

CString message;

pDlg->m\_tx\_edit.GetWindowText(message);//입력받은 메세지를 얻어서

//message += "\r\n";

pDlg->m\_tx\_edit.SetWindowTextW(message);// 메세지를 출력

pDlg->m\_pDataSocket->Send(str, (str.GetLength() + 1) \* sizeof(TCHAR));

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount());

plist->RemoveAt(current\_pos); // 현재위치의 plist 삭제

}

Sleep(10);

}

return 0;

}

// CTCPClient\_thdDlg 메시지 처리기

BOOL CTCPClient\_thdDlg::OnInitDialog()

{

CDialogEx::OnInitDialog();

// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

// IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

BOOL bNameValid;

CString strAboutMenu;

bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);

ASSERT(bNameValid);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는

// 프레임워크가 이 작업을 자동으로 수행합니다.

SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.

SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.

// TODO: 여기에 추가 초기화 작업을 추가합니다.

m\_pDataSocket = NULL;

m\_ipaddr.SetWindowTextW(\_T("127.0.0.1"));

//TX queue

CStringList\* newlist1 = new CStringList;

arg1.pList = newlist1;

arg1.Thread\_run = 1; //스레드를 실행 상태로 표시

arg1.pDlg = this;

//RX queue

CStringList\* newlist2 = new CStringList;

arg2.pList = newlist2;

arg2.Thread\_run = 1; //스레드를 실행 상태로 표시

arg2.pDlg = this;

pThread1 = AfxBeginThread(TXThread, (LPVOID)&arg1);

pThread2 = AfxBeginThread(RXThread, (LPVOID)&arg2);

return TRUE; // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.

}

void CTCPClient\_thdDlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialogEx::OnSysCommand(nID, lParam);

}

}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면

// 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는

// 프레임워크에서 이 작업을 자동으로 수행합니다.

void CTCPClient\_thdDlg::OnPaint()

{

if (IsIconic())

{

CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);

// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.

int cxIcon = GetSystemMetrics(SM\_CXICON);

int cyIcon = GetSystemMetrics(SM\_CYICON);

CRect rect;

GetClientRect(&rect);

int x = (rect.Width() - cxIcon + 1) / 2;

int y = (rect.Height() - cyIcon + 1) / 2;

// 아이콘을 그립니다.

dc.DrawIcon(x, y, m\_hIcon);

}

else

{

CDialogEx::OnPaint();

}

}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서

// 이 함수를 호출합니다.

HCURSOR CTCPClient\_thdDlg::OnQueryDragIcon()

{

return static\_cast<HCURSOR>(m\_hIcon);

}

void CTCPClient\_thdDlg::ProcessReceive(CDataSocket \*pSocket, int nErrorCode)

{

TCHAR pBuf[1024 + 1];

CString strData;

int nbytes=0;

nbytes = pSocket->Receive(pBuf, 1024);

pBuf[nbytes] = NULL;

strData = (LPCTSTR)pBuf;

rx\_cs.Lock();

arg2.pList->AddTail((LPCTSTR)strData);

rx\_cs.Unlock();

}

void CTCPClient\_thdDlg::OnBnClickedConnect()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL) ////////

{

m\_pDataSocket = new CDataSocket(this);

m\_pDataSocket->Create();

CString addr;

m\_ipaddr.GetWindowText(addr);

if (m\_pDataSocket->Connect(addr, 8000)) //읽어온 주소로 연결되면

{

//MessageBox(\_T("서버에 접속 성공!"), \_T("알림"), MB\_ICONINFORMATION);

CString message = \_T("### 서버에 접속 성공! ###\r\n");

m\_rx\_edit.SetWindowTextW(message);

}

else

{

//MessageBox(\_T("서버에 접속 실패!"), \_T("알림"), MB\_ICONERROR);

CString message = \_T("### 서버에 접속 실패! ###\n");

m\_rx\_edit.SetWindowTextW(message);

delete m\_pDataSocket;

m\_pDataSocket = NULL;

}

}

else

{

MessageBox(\_T("서버에 이미 접속됨!"), \_T("알림"), MB\_ICONINFORMATION);

//m\_rx\_edit.SetFocus(); ////////

}

}

void CTCPClient\_thdDlg::OnBnClickedDisconnect()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL)

{

MessageBox(\_T("서버에 접속 안 함!"), \_T("알림"), MB\_ICONINFORMATION);

}

else

{

arg1.Thread\_run = 0;

arg2.Thread\_run = 0;

m\_pDataSocket->Close();

delete m\_pDataSocket;

m\_pDataSocket = NULL;

}

}

void CTCPClient\_thdDlg::OnBnClickedSend()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL)

{

MessageBox(\_T("서버에 접속 안 함!"), \_T("알림"), MB\_ICONERROR);

}

else

{

CString message;

m\_tx\_edit\_short.GetWindowText(message); //텍스트 받고

message += \_T("\r\n");

tx\_cs.Lock();

arg1.pList->AddTail(message); //메시지 담고

tx\_cs.Unlock();

m\_tx\_edit\_short.SetWindowTextW(\_T(""));

m\_tx\_edit\_short.SetFocus();

int len = m\_tx\_edit.GetWindowTextLengthW();

m\_tx\_edit.SetSel(len, len);

m\_tx\_edit.ReplaceSel(message); //받은 메시지 큰 edit창에 출력

}

}

[Lab10] UDP 양방향 통신

UDPServer\_thdDlg.cpp

// CUDPServer\_thdDlg 메시지 처리기

UINT RXThread(LPVOID arg)

{

ThreadArg \*pArg = (ThreadArg\*)arg;

CStringList \*plist = pArg->pList;

CUDPServer\_thdDlg \*pDlg = (CUDPServer\_thdDlg \*)pArg->pDlg;

while (pArg->Thread\_run) //스레드가 실행상태일동안

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

current\_pos = pos;

rx\_cs.Lock();

CString str = plist->GetNext(pos);

rx\_cs.Unlock();

CString message;

pDlg->m\_rx\_edit.GetWindowText(message);

message += str;

pDlg->m\_rx\_edit.SetWindowTextW(message);

pDlg->m\_rx\_edit.LineScroll(pDlg->m\_rx\_edit.GetLineCount());

plist->RemoveAt(current\_pos);

}

Sleep(10);

}

return 0;

}

UINT TXThread(LPVOID arg)

{

//////////////

ThreadArg \*pArg = (ThreadArg \*)arg; // ThreadArg 객체 생성

CStringList \*plist = pArg->pList;

CUDPServer\_thdDlg \*pDlg = (CUDPServer\_thdDlg\*)pArg->pDlg;

CString rSocketAddr = \_T("127.0.0.1");

UINT nHostPort = 5000; ///////////////상대편 포트 번호

while (pArg->Thread\_run) //스레드가 run상태일때만 동작

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL) {

current\_pos = pos;

rx\_cs.Lock();

CString str = plist->GetNext(pos);

rx\_cs.Unlock();

CString message;

pDlg->m\_tx\_edit.GetWindowText(message); //입력받은 메세지를 얻어서

//message += "\r\n";

pDlg->m\_tx\_edit.SetWindowTextW(message); // 메세지를 출력

pDlg->m\_pDataSocket->SendToEx(str, (str.GetLength() + 1) \* sizeof(TCHAR), nHostPort, rSocketAddr, 0); ////UDP 패킷 전송

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount());

plist->RemoveAt(current\_pos); // 현재위치의 plist 삭제

}

Sleep(10);

}

return 0;

}

BOOL CUDPServer\_thdDlg::OnInitDialog()

{

CDialogEx::OnInitDialog();

// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

// IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

BOOL bNameValid;

CString strAboutMenu;

bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);

ASSERT(bNameValid);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는

// 프레임워크가 이 작업을 자동으로 수행합니다.

SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.

SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.

// TODO: 여기에 추가 초기화 작업을 추가합니다.

m\_pDataSocket = new CDataSocket(this);

m\_pDataSocket->Create(8000, SOCK\_DGRAM); //Create() 멤버함수 사용해서 UDP 소켓 생성

m\_pDataSocket->Connect(\_T("127.0.0.1"), 5000);/////상대편포트번호로 연결

//TX queue

CStringList \*newlist1 = new CStringList;

arg1.pList = newlist1;

arg1.Thread\_run = 1; //스레드를 실행 상태로 표시

arg1.pDlg = this;

//RX queue

CStringList\* newlist2 = new CStringList;

arg2.pList = newlist2;

arg2.Thread\_run = 1; //스레드를 실행 상태로 표시

arg2.pDlg = this;

/\* 윈도우 소켓 초기화 \*/

WSADATA wsa;

int error\_code;

if ((error\_code = WSAStartup(MAKEWORD(2, 2), &wsa)) != 0)

{

TCHAR buffer[256];

FormatMessageW(FORMAT\_MESSAGE\_FROM\_SYSTEM, NULL, error\_code,

MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT), buffer, 256, NULL);

AfxMessageBox(buffer, MB\_ICONERROR);

}

pThread1 = AfxBeginThread(TXThread, (LPVOID)&arg1);

pThread2 = AfxBeginThread(RXThread, (LPVOID)&arg2);

return FALSE;

}

void CUDPServer\_thdDlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialogEx::OnSysCommand(nID, lParam);

}

}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면

// 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는

// 프레임워크에서 이 작업을 자동으로 수행합니다.

void CUDPServer\_thdDlg::OnPaint()

{

if (IsIconic())

{

CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);

// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.

int cxIcon = GetSystemMetrics(SM\_CXICON);

int cyIcon = GetSystemMetrics(SM\_CYICON);

CRect rect;

GetClientRect(&rect);

int x = (rect.Width() - cxIcon + 1) / 2;

int y = (rect.Height() - cyIcon + 1) / 2;

// 아이콘을 그립니다.

dc.DrawIcon(x, y, m\_hIcon);

}

else

{

CDialogEx::OnPaint();

}

}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서

// 이 함수를 호출합니다.

HCURSOR CUDPServer\_thdDlg::OnQueryDragIcon()

{

return static\_cast<HCURSOR>(m\_hIcon);

}

void CUDPServer\_thdDlg::OnBnClickedSend()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

CString message;

m\_tx\_edit\_short.GetWindowText(message); //텍스트 받고

message += \_T("\r\n"); //줄바꿈 위해

tx\_cs.Lock();

arg1.pList->AddTail(message);//리스트에 메시지 담고

tx\_cs.Unlock();

m\_tx\_edit\_short.SetWindowTextW(\_T("")); //입력했던 작은 edit창의 텍스트 지울 수 있도록

m\_tx\_edit\_short.SetFocus();

int len = m\_tx\_edit.GetWindowTextLengthW();

m\_tx\_edit.SetSel(len, len);

m\_tx\_edit.ReplaceSel(message); //받은 메시지 큰 edit창에 출력

}

void CUDPServer\_thdDlg::OnBnClickedClose()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL)

{

AfxMessageBox(\_T("이미 접속 종료"));

}

else

{

MessageBox(\_T("접속 종료"), \_T("알림"), MB\_ICONINFORMATION);

/\* 스레드를 종료 상태로 명시 \*/

arg1.Thread\_run = 0;

arg2.Thread\_run = 0;

m\_pDataSocket->Close();

delete m\_pDataSocket;

m\_pDataSocket = NULL;

/\*m\_pListenSocket->Close();

delete m\_pListenSocket;

m\_pListenSocket = NULL;\*/

}

}

void CUDPServer\_thdDlg::ProcessReceive(CDataSocket \*pSocket, int nErrorCode)

{

TCHAR pBuf[1024 + 1];

CString strData;

int nbytes = 0;

CString rSocketAddr = \_T("127.0.0.1");

UINT rSocketPort = 8000; //서버포트번호

nbytes = pSocket->ReceiveFromEx(pBuf, 1024, rSocketAddr, rSocketPort, 0); //UDP 패킷 수신

pBuf[nbytes] = NULL;

strData = (LPCTSTR)pBuf;

rx\_cs.Lock();

arg2.pList->AddTail((LPCTSTR)strData);

rx\_cs.Unlock();

}

UDPClient\_thdDlg.cpp

UINT RXThread(LPVOID arg)

{

ThreadArg \*pArg = (ThreadArg\*)arg;

CStringList \*plist = pArg->pList;

CUDPClient\_thdDlg \*pDlg = (CUDPClient\_thdDlg \*)pArg->pDlg;

while (pArg->Thread\_run)//스레드가 실행상태일동안

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

current\_pos = pos;

rx\_cs.Lock();

CString str = plist->GetNext(pos);

rx\_cs.Unlock();

CString message;

pDlg->m\_rx\_edit.GetWindowText(message);

message += str;

pDlg->m\_rx\_edit.SetWindowTextW(message);

pDlg->m\_rx\_edit.LineScroll(pDlg->m\_rx\_edit.GetLineCount());

plist->RemoveAt(current\_pos);

}

Sleep(10);

}

return 0;

}

UINT TXThread(LPVOID arg)

{

ThreadArg \*pArg = (ThreadArg \*)arg;

CStringList \*plist = pArg->pList;

CUDPClient\_thdDlg \*pDlg = (CUDPClient\_thdDlg\*)pArg->pDlg;

CString rSocketAddr = \_T("127.0.0.1");

UINT nHostPort = 8000; //상대편 포트 번호

while (pArg->Thread\_run) // 스레드가 run상태일때만 동작

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL) {

current\_pos = pos;

tx\_cs.Lock();

CString str = plist->GetNext(pos);

tx\_cs.Unlock();

CString message;

pDlg->m\_tx\_edit.GetWindowText(message);//입력받은 메세지를 얻어서

pDlg->m\_tx\_edit.SetWindowTextW(message);// 메세지를 출력

pDlg->m\_pDataSocket->SendToEx(str, (str.GetLength() + 1) \* sizeof(TCHAR), nHostPort, rSocketAddr, 0); //UDP 패킷 전송

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount());

plist->RemoveAt(current\_pos); // 현재위치의 plist 삭제

}

Sleep(10);

}

return 0;

}

// CUDPClient\_thdDlg 메시지 처리기

BOOL CUDPClient\_thdDlg::OnInitDialog()

{

CDialogEx::OnInitDialog();

// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

// IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

BOOL bNameValid;

CString strAboutMenu;

bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);

ASSERT(bNameValid);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는

// 프레임워크가 이 작업을 자동으로 수행합니다.

SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.

SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.

// TODO: 여기에 추가 초기화 작업을 추가합니다.

m\_pDataSocket = NULL;

m\_pDataSocket = new CDataSocket(this);

m\_pDataSocket->Create(5000, SOCK\_DGRAM); //Create() 멤버함수 사용해서 UDP 소켓 생성

m\_pDataSocket->Connect(\_T("127.0.0.1"), 8000); //상대편포트번호로 연결

m\_ipaddr.SetWindowTextW(\_T("127.0.0.1")); //IP주소 텍스트 설정

//TX queue

CStringList\* newlist1 = new CStringList;

arg1.pList = newlist1;

arg1.Thread\_run = 1; //스레드를 실행 상태로 표시

arg1.pDlg = this;

//RX queue

CStringList\* newlist2 = new CStringList;

arg2.pList = newlist2;

arg2.Thread\_run = 1; //스레드를 실행 상태로 표시

arg2.pDlg = this;

pThread1 = AfxBeginThread(TXThread, (LPVOID)&arg1);

pThread2 = AfxBeginThread(RXThread, (LPVOID)&arg2);

return TRUE; // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.

}

void CUDPClient\_thdDlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialogEx::OnSysCommand(nID, lParam);

}

}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면

// 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는

// 프레임워크에서 이 작업을 자동으로 수행합니다.

void CUDPClient\_thdDlg::OnPaint()

{

if (IsIconic())

{

CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);

// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.

int cxIcon = GetSystemMetrics(SM\_CXICON);

int cyIcon = GetSystemMetrics(SM\_CYICON);

CRect rect;

GetClientRect(&rect);

int x = (rect.Width() - cxIcon + 1) / 2;

int y = (rect.Height() - cyIcon + 1) / 2;

// 아이콘을 그립니다.

dc.DrawIcon(x, y, m\_hIcon);

}

else

{

CDialogEx::OnPaint();

}

}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서

// 이 함수를 호출합니다.

HCURSOR CUDPClient\_thdDlg::OnQueryDragIcon()

{

return static\_cast<HCURSOR>(m\_hIcon);

}

void CUDPClient\_thdDlg::OnBnClickedSend()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL)

{

MessageBox(\_T("서버에 접속 안 함!"), \_T("알림"), MB\_ICONERROR);

}

else

{

CString message;

m\_tx\_edit\_short.GetWindowText(message); //텍스트 받고

message += \_T("\r\n");

tx\_cs.Lock();

arg1.pList->AddTail(message); //리스트에 메시지 담고

tx\_cs.Unlock();

m\_tx\_edit\_short.SetWindowTextW(\_T("")); //입력했던 작은 edit창의 텍스트 지울 수 있도록

m\_tx\_edit\_short.SetFocus();

int len = m\_tx\_edit.GetWindowTextLengthW();

m\_tx\_edit.SetSel(len, len);

m\_tx\_edit.ReplaceSel(message); //받은 메시지 큰 edit창에 출력

}

}

void CUDPClient\_thdDlg::OnBnClickedClose()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL)

{

MessageBox(\_T("서버에 접속 안 함!"), \_T("알림"), MB\_ICONINFORMATION);

}

else

{

MessageBox(\_T("접속 종료"), \_T("알림"), MB\_ICONINFORMATION);

/\* 스레드를 종료 상태로 명시 \*/

arg1.Thread\_run = 0;

arg2.Thread\_run = 0;

m\_pDataSocket->Close(); //Close멤버함수로 소켓 종ㄹ료

delete m\_pDataSocket; //삭제

m\_pDataSocket = NULL;

}

}

void CUDPClient\_thdDlg::ProcessReceive(CDataSocket \*pSocket, int nErrorCode)

{

TCHAR pBuf[1024 + 1];

CString strData;

int nbytes = 0;

CString rSocketAddr = \_T("127.0.0.1");

UINT rSocketPort = 8000;

nbytes = pSocket->ReceiveFromEx(pBuf, 1024, rSocketAddr, rSocketPort, 0); //UDP 패킷 수신

pBuf[nbytes] = NULL;

strData = (LPCTSTR)pBuf;

rx\_cs.Lock();

arg2.pList->AddTail((LPCTSTR)strData);

rx\_cs.Unlock();

}

[마지막실습과제]

1. UDPServer부분

// UDPServer\_thdDlg.cpp : 구현 파일

//

#include "stdafx.h"

#include "UDPServer\_thd.h"

#include "UDPServer\_thdDlg.h"

#include "afxdialogex.h"

#include "DataSocket.h"

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

//////////////////////////////////////////////////////////

CCriticalSection tx\_cs;

CCriticalSection rx\_cs;

int sq = 0;

CWinThread \*pTime\_th;

//////////////////////////////////////////////////////////

// 응용 프로그램 정보에 사용되는 CAboutDlg 대화 상자입니다.

class CAboutDlg : public CDialogEx

{

public:

CAboutDlg();

// 대화 상자 데이터입니다.

#ifdef AFX\_DESIGN\_TIME

enum { IDD = IDD\_ABOUTBOX };

#endif

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 지원입니다.

// 구현입니다.

protected:

DECLARE\_MESSAGE\_MAP()

};

CAboutDlg::CAboutDlg() : CDialogEx(IDD\_ABOUTBOX)

{

}

void CAboutDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

}

BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)

END\_MESSAGE\_MAP()

// CUDPServer\_thdDlg 대화 상자

CUDPServer\_thdDlg::CUDPServer\_thdDlg(CWnd\* pParent /\*=NULL\*/)

: CDialogEx(IDD\_UDPSERVER\_THD\_DIALOG, pParent)

{

m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);

}

void CUDPServer\_thdDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Control(pDX, IDC\_EDIT1, m\_tx\_edit\_short);

DDX\_Control(pDX, IDC\_EDIT3, m\_tx\_edit);

// DDX\_Control(pDX, IDC\_EDIT2, m\_rx\_edit);

DDX\_Control(pDX, IDC\_EDIT4, m\_tx\_Thread);

DDX\_Control(pDX, IDC\_EDIT5, m\_rx\_Thread);

}

BEGIN\_MESSAGE\_MAP(CUDPServer\_thdDlg, CDialogEx)

ON\_WM\_SYSCOMMAND()

ON\_WM\_PAINT()

ON\_BN\_CLICKED(IDC\_SEND, &CUDPServer\_thdDlg::OnBnClickedSend)

ON\_BN\_CLICKED(IDC\_CLOSE, &CUDPServer\_thdDlg::OnBnClickedClose)

ON\_EN\_CHANGE(IDC\_EDIT3, &CUDPServer\_thdDlg::OnEnChangeEdit3)

END\_MESSAGE\_MAP()

// CUDPServer\_thdDlg 메시지 처리기

/////////////////////////////////////////////////////////////

static CString Buf = \_T("");

UINT RXThread(LPVOID arg)//Receive (Rx)

{

ThreadArg\_Receive \*pArg = (ThreadArg\_Receive \*)arg; //스레드 인자 포인터 pArg생성하여 매개변수 arg를 대입한다.

CList<Frame> \*plist = pArg->pStringList; //리스트 포인터 plist를 생성하여 pArg->pList를 대입한다.

CUDPServer\_thdDlg \*pDlg = (CUDPServer\_thdDlg \*)pArg->pDlg; //dlg포인터 pDlg생성하여 pArg->pList를 대입한다.

UINT port = 8001; CString addr = \_T("127.0.0.1");

CStringList \* stringlist;

while (pArg->Thread\_run)

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

Frame ack\_frame;

current\_pos = pos;

rx\_cs.Lock(); //Critical Section method Lock()을 다른스레드가 접근 못 하게 함

Frame frame = plist->GetNext(pos); //pos의 다음위치를 plist가 가르키게하고 그 값을 str변수에 저장시켜준다.

rx\_cs.Unlock(); //다른 스레드의 접근을 가능하게 한다.

unsigned short kk = Checksum((unsigned short\*)&frame, sizeof(Frame));

if (kk == 0)//CHECKSUM OK

{

CString temp;

ack\_frame.sqN = frame.sqN + 1;

ack\_frame.flag\_s = ACK;

rx\_cs.Lock();

pDlg->m\_pDataSocket->SendToEx(&ack\_frame, sizeof(Frame), port, addr);

rx\_cs.Unlock();

temp.Format(\_T("seqNum: %d, flag\_s :%c\r\n"), (int)ack\_frame.sqN, ack\_frame.flag\_s);

pDlg->m\_tx\_Thread.SetWindowTextW(temp);

stringlist = new CStringList;

stringlist->AddTail((CString)frame.data);

int len = pDlg->m\_rx\_Thread.GetWindowTextLengthW();

pDlg->m\_rx\_Thread.SetSel(len, len);

temp.Format(\_T("seqNum:%d, checksum:%d, flag\_s:%c\r\n"), (int)frame.sqN, frame.checkSum, frame.flag\_s);

pDlg->m\_rx\_Thread.ReplaceSel(temp);

CString message;

pDlg->m\_tx\_edit.GetWindowText(message); //dialoge에 접근하여 창에 쓰여진 message를 얻어옴

POSITION fpos = stringlist->GetHeadPosition();

POSITION fcurrent\_pos;

while (fpos != NULL)

{

fcurrent\_pos = fpos;

CString str = stringlist->GetNext(fpos);

//AfxMessageBox(str);

if ((frame.end\_game != '`') && (frame.end\_game != '/') && (str.GetLength() <= 16))

{

str = \_T("서버 : ") + str;

message += str + \_T("\r\n");

message += "\n";

}

if ((frame.end\_game == '`'))

{

if (str.GetLength() == 17)

{

str = str.Mid(0, 16);

}

Buf = Buf + str;

//AfxMessageBox(Buf);

}

if (frame.end\_game == '/')

{

Buf = Buf + str;

//AfxMessageBox(Buf);

str = Buf;

Buf = \_T("");

str.Remove('`'); str.Remove('/');

str = \_T("클라 : ") + str;

message += str + \_T("\r\n");

message += "\n";

}

stringlist->RemoveAt(fcurrent\_pos);

}

pDlg->m\_tx\_edit.SetWindowText(message); //메세지를 창에 출력한다

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount()); //텍스트 스크롤

plist->RemoveAt(current\_pos); //리스트에서 꺼낸 데이터 삭제

}

else

{

ack\_frame.sqN = frame.sqN + 1;

ack\_frame.flag\_s = NACK;

rx\_cs.Lock();

pDlg->m\_pDataSocket->SendToEx(&ack\_frame, sizeof(Frame), port, addr);

rx\_cs.Unlock();

}

}

Sleep(10);

}

return 0;

}

UINT TXThread(LPVOID arg) //보내는 부분, 따라하기 외 구현한 부분//////////////////////////////

{

ThreadArg\_Send \*pArg = (ThreadArg\_Send \*)arg; //ThreadArg 객체

CStringList \*plist = pArg->pList;

CUDPServer\_thdDlg \*pDlg = (CUDPServer\_thdDlg\*)pArg->pDlg;

CString message;

pDlg->m\_tx\_edit.GetWindowTextW(message);

CString SocketAddress = \_T("127.0.0.1"); //addr

UINT SocketPort = 8001; //Port

CStringList \* stringlist;

CList<Frame> \*alist = pArg->pAckList;

CList<Frame> \*ack\_list;

while (pArg->Thread\_run) {

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

Frame temp;

memset(&temp, 0, sizeof(temp));

current\_pos = pos;

tx\_cs.Lock();

CString str = plist->GetNext(pos);

tx\_cs.Unlock();

stringlist = new CStringList;

stringlist->AddTail(str);

int length;

int length\_w = str.GetLength();

CString message;

pDlg->m\_tx\_edit.GetWindowTextW(message);

pDlg->m\_tx\_edit.SetWindowText(message);

if (str.Find('`') >=0 || str.GetLength() == 17)

{

str.Remove('`');

temp.end\_game = '`';

}

if (str.Find('/') >= 0)

{

str.Remove('/');

temp.end\_game = '/';

}

wchar\_t\* dataW = new wchar\_t[length\_w];

dataW = (wchar\_t\*)(str.GetString());

length = WideCharToMultiByte(CP\_ACP, 0, dataW, -1, NULL, 0, NULL, NULL);

char \*dataMB = new char[length];

memset(dataMB, 0, sizeof(dataMB));

WideCharToMultiByte(CP\_ACP, 0, dataW, -1, dataMB, length, 0, 0);

/////////////////////////////////

memcpy(temp.data, dataMB, 17);////

////////////////////////////////

temp.sqN = sq++;

temp.flag\_s = SEND;

temp.checkSum = Checksum((unsigned short \*)&temp, sizeof(Frame));

CString asd;

asd = temp.data;

AfxMessageBox(asd);

CString tx;

tx.Format(\_T("seqNum:%d, checksum:%d, flag\_s:%c, end : %c\r\n"), (int)temp.sqN, temp.checkSum, temp.flag\_s, temp.end\_game);

CString TXmessage;

pDlg->m\_tx\_edit.GetWindowText(TXmessage);

pDlg->m\_pDataSocket->SendToEx(&temp, sizeof(temp), SocketPort, SocketAddress);

int len = pDlg->m\_tx\_Thread.GetWindowTextLengthW();

pDlg->m\_tx\_Thread.SetSel(len, len);

pDlg->m\_tx\_Thread.ReplaceSel(tx);

pDlg->m\_tx\_Thread.SetWindowTextW(tx);

pDlg->m\_tx\_Thread.LineScroll(pDlg->m\_tx\_Thread.GetLineCount());

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount());

Sleep(30);

char status = ACK;

POSITION ack\_Pos = alist->GetHeadPosition();

POSITION ack\_Current\_pos;

bool flag\_t = 0;

pTime\_th = AfxBeginThread(TimeOutCheckThread, (LPVOID)&flag\_t);

if (flag\_t != 1)

{

//ASSERT(0);

if (ack\_Pos == NULL)

AfxMessageBox(\_T("액크 리스트 널이다"));

if (ack\_Pos != NULL)

{

ack\_Current\_pos = ack\_Pos;

Frame ack\_temp = alist->GetNext(ack\_Pos);

CString ack\_txt;

ack\_txt.Format(\_T("Receive->ACK Sq : %d flag : %c"), ack\_temp.sqN, ack\_temp.flag\_s);

pDlg->m\_rx\_Thread.SetWindowTextW(ack\_txt);

status = ack\_temp.flag\_s;

alist->RemoveAt(ack\_Current\_pos);

}

}

if (flag\_t == 1||status == NACK)

{

temp.flag\_s = RESEND;

tx\_cs.Lock();

pDlg->m\_pDataSocket->SendToEx(&temp, sizeof(temp), SocketPort, SocketAddress);

tx\_cs.Unlock();

}

plist->RemoveAt(current\_pos);

}

Sleep(10);

}

return 0;

}

//////////////////////////////////////////////////////////////

BOOL CUDPServer\_thdDlg::OnInitDialog()

{

CDialogEx::OnInitDialog();

// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

// IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

BOOL bNameValid;

CString strAboutMenu;

bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);

ASSERT(bNameValid);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는

// 프레임워크가 이 작업을 자동으로 수행합니다.

SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.

SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.

// TODO: 여기에 추가 초기화 작업을 추가합니다.

m\_pDataSocket = new CDataSocket(this);

CStringList\* strList = new CStringList;

arg1.pList = strList;

arg1.Thread\_run = 1;

arg1.pDlg = this;

CList<Frame>\* ackList1 = new CList<Frame>;

arg1.pAckList = ackList1;

CList<Frame>\* frameList = new CList<Frame>;

arg2.pStringList = frameList;

arg2.Thread\_run = 1;

arg2.pDlg = this;

CList<Frame>\* ackList2 = new CList<Frame>;

arg2.pAckList = ackList2;

WSADATA wsa;

int error\_code;

if ((error\_code = WSAStartup(MAKEWORD(2, 2), &wsa)) != 0)

{

TCHAR buffer[256];

FormatMessageW(FORMAT\_MESSAGE\_FROM\_SYSTEM, NULL, error\_code, MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT), buffer, 256, NULL);

AfxMessageBox(buffer, MB\_ICONERROR);

}

if (m\_pDataSocket->Create(8000, SOCK\_DGRAM)) {

pThread1 = AfxBeginThread(TXThread, (LPVOID)&arg1); //스레드를 생성시키고, 그것을 pThread1이라고한다.

pThread2 = AfxBeginThread(RXThread, (LPVOID)&arg2); //스레드를 생성시키고, 그것을 pThread2라고 한다.

return TRUE;

}

else {

int err = m\_pDataSocket->GetLastError();

TCHAR buffer[256];

FormatMessageW(FORMAT\_MESSAGE\_FROM\_SYSTEM, NULL, error\_code,

MAKELANGID(LANG\_NEUTRAL, SUBLANG\_DEFAULT), buffer, 256, NULL);

AfxMessageBox(buffer, MB\_ICONERROR);

}

AfxMessageBox(\_T("이미 실행 중인 서버가 있습니다")\_T("\n프로그램을 종료합니다"), MB\_ICONERROR);

return FALSE;

}

void CUDPServer\_thdDlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialogEx::OnSysCommand(nID, lParam);

}

}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면

// 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는

// 프레임워크에서 이 작업을 자동으로 수행합니다.

void CUDPServer\_thdDlg::OnPaint()

{

if (IsIconic())

{

CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);

// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.

int cxIcon = GetSystemMetrics(SM\_CXICON);

int cyIcon = GetSystemMetrics(SM\_CYICON);

CRect rect;

GetClientRect(&rect);

int x = (rect.Width() - cxIcon + 1) / 2;

int y = (rect.Height() - cyIcon + 1) / 2;

// 아이콘을 그립니다.

dc.DrawIcon(x, y, m\_hIcon);

}

else

{

CDialogEx::OnPaint();

}

}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서

// 이 함수를 호출합니다.

HCURSOR CUDPServer\_thdDlg::OnQueryDragIcon()

{

return static\_cast<HCURSOR>(m\_hIcon);

}

void CUDPServer\_thdDlg::ProcessReceive(CDataSocket\* pSocket, int nErororCode)

{

Frame get;

Frame\* Data = &get;

int nbytes;

CString SocketAddress = \_T("127.0.0.1");

UINT SocketPort = 8001;

nbytes = pSocket->ReceiveFromEx(&get, sizeof(Frame), SocketAddress, SocketPort, 0);

if (get.flag\_s == ACK || get.flag\_s == NACK)

{

tx\_cs.Lock();

arg1.pAckList->AddTail(\*Data);

CString temp;

temp.Format(\_T("AckFrame's Seq.No : %d, Flag\_s : %c"), get.sqN, get.flag\_s);

AfxMessageBox(temp);

tx\_cs.Unlock();

}

else if (get.sqN == SEND || get.flag\_s == RESEND)

{

rx\_cs.Lock();

arg2.pStringList->AddTail(\*Data);

rx\_cs.Unlock();

}

else {

rx\_cs.Lock();

arg2.pStringList->AddTail(\*Data);

arg2.pAckList->AddTail(\*Data);

rx\_cs.Unlock();

}

}

void CUDPServer\_thdDlg::ProcessClose(CDataSocket\* pSocket, int nErrorCode)

{

pSocket->Close();

delete m\_pDataSocket;

m\_pDataSocket = NULL;

int len = m\_rx\_edit.GetWindowTextLengthW();

CString message = \_T("###접속 종료 ###\r\n\r\n");

m\_rx\_edit.SetSel(len, len);

m\_rx\_edit.ReplaceSel(message);

}

void CUDPServer\_thdDlg::OnBnClickedSend()

{

CString tx\_Smessage = NULL;

m\_tx\_edit\_short.GetWindowText(tx\_Smessage);

int length = 0;

CString \*temp;

length = tx\_Smessage.GetLength();

if ((length <= 16)) { //만약에 문자열길이가 16바이트이거나

//또는 16바이트 보다 작으면 길이를 1로 주어지게.

CString \*temp = new CString[1];

temp[0] = tx\_Smessage.Mid(0, 16);

tx\_cs.Lock();

arg1.pList->AddTail(temp[0]);

tx\_cs.Unlock();

}

else if ((length > 16) && (length % 16) != 0) { //만약 문자열 길이가 16보다 클 경우 길이가

CString \*temp = new CString[(length / 16) + 1];

for (int i = 0; i < (length / 16) + 1; i++)

{

if (i == (length / 16))

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("/");

else

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("`");

}

tx\_cs.Lock();

for (int i = 0; i < (length / 16) + 1; i++) {

arg1.pList->AddTail(temp[i]);

}

tx\_cs.Unlock();

}

else {

CString \*temp = new CString[(length / 16) + 1];

for (int i = 0; i < (length / 16) + 1; i++)

{

if (i == (length / 16))

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("/");

else

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("`");

}

tx\_cs.Lock();

for (int i = 0; i < (length / 16) + 1; i++) {

arg1.pList->AddTail(temp[i]);

}

tx\_cs.Unlock();

}

m\_tx\_edit\_short.SetWindowText(\_T(""));//m\_tx\_edit\_short 비우기

m\_tx\_edit\_short.SetFocus();//입력 상태로 만들기

tx\_Smessage = \_T("서버 : ") + tx\_Smessage;

tx\_Smessage += "\r\n";

int len = m\_tx\_edit.GetWindowTextLengthW();

m\_tx\_edit.SetSel(len, len);

m\_tx\_edit.ReplaceSel(tx\_Smessage);//보낸 tx\_Smessage 띄우기

}

void CUDPServer\_thdDlg::OnBnClickedClose()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL) {

AfxMessageBox(\_T("이미 접속 종료"));

}

else

{

arg1.Thread\_run = 0;

arg2.Thread\_run = 0;

m\_pDataSocket->Close();

delete m\_pDataSocket;

m\_pDataSocket = NULL;

}

}

BOOL CUDPServer\_thdDlg::PreTranslateMessage(MSG \* pMsg)

{

return CDialogEx::PreTranslateMessage(pMsg);

if (pMsg->message == WM\_KEYDOWN)

{

if (pMsg->wParam == VK\_RETURN)

{

OnBnClickedSend();

//return true;

}

else if (pMsg->wParam == VK\_ESCAPE)

{

OnOK();

//return true;

}

}

}

void CUDPServer\_thdDlg::OnEnChangeEdit3()

{

// TODO: RICHEDIT 컨트롤인 경우, 이 컨트롤은

// CDialogEx::OnInitDialog() 함수를 재지정

//하고 마스크에 OR 연산하여 설정된 ENM\_CHANGE 플래그를 지정하여 CRichEditCtrl().SetEventMask()를 호출하지 않으면

// 이 알림 메시지를 보내지 않습니다.

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

}

unsigned short Checksum(unsigned short \*ptr, int size)

{

Frame \*frame = (Frame \*)ptr; //checksum을 생성할 문자

unsigned short checksum = 0;

unsigned short oddbyte;

register short result;

while (size > 1)

{

checksum += \*ptr++;

size -= 2;

}

if (size == 1)

{

oddbyte = 0;

\*((u\_char\*)&oddbyte) = \*((u\_char\*)ptr);//oddbyte가 가리키는 주소에 들어있는 거에 ptr이 가리키는 내용을 넣음

checksum += oddbyte;

}

checksum = (checksum >> 16) + checksum; //carry처리

checksum = ~checksum; //compliment

return checksum;

}

UINT TimeOutCheckThread(LPVOID arg)

{

int \*flag\_t = (int \*)arg;

Sleep(5000);

(\*flag\_t) = 1 - (\*flag\_t);

//\*(bool\*)arg = false;

return 0;

}

2.UDPClinet부분

// UDPClient\_thdDlg.cpp : 구현 파일

//

#include "stdafx.h"

#include "UDPClient\_thd.h"

#include "UDPClient\_thdDlg.h"

#include "afxdialogex.h"

#include "DataSocket.h"

#ifdef \_DEBUG

#define new DEBUG\_NEW

#endif

////////////////////////////////////////////

CCriticalSection tx\_cs;

CCriticalSection rx\_cs;

CWinThread \*pTime\_th;

int sq=0;

/////////////////////////////////////////////

// 응용 프로그램 정보에 사용되는 CAboutDlg 대화 상자입니다.

class CAboutDlg : public CDialogEx

{

public:

CAboutDlg();

// 대화 상자 데이터입니다.

#ifdef AFX\_DESIGN\_TIME

enum { IDD = IDD\_ABOUTBOX };

#endif

protected:

virtual void DoDataExchange(CDataExchange\* pDX); // DDX/DDV 지원입니다.

// 구현입니다.

protected:

DECLARE\_MESSAGE\_MAP()

};

CAboutDlg::CAboutDlg() : CDialogEx(IDD\_ABOUTBOX)

{

}

void CAboutDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

}

BEGIN\_MESSAGE\_MAP(CAboutDlg, CDialogEx)

END\_MESSAGE\_MAP()

// CUDPClient\_thdDlg 대화 상자

CUDPClient\_thdDlg::CUDPClient\_thdDlg(CWnd\* pParent /\*=NULL\*/)

: CDialogEx(IDD\_UDPCLIENT\_THD\_DIALOG, pParent)

{

m\_hIcon = AfxGetApp()->LoadIcon(IDR\_MAINFRAME);

}

void CUDPClient\_thdDlg::DoDataExchange(CDataExchange\* pDX)

{

CDialogEx::DoDataExchange(pDX);

DDX\_Control(pDX, IDC\_IPADDRESS1, m\_ipaddr);

DDX\_Control(pDX, IDC\_EDIT1, m\_tx\_edit\_short);

DDX\_Control(pDX, IDC\_EDIT3, m\_tx\_edit);

//DDX\_Control(pDX, IDC\_EDIT2, m\_rx\_edit);

DDX\_Control(pDX, IDC\_EDIT4, m\_tx\_Thread);

DDX\_Control(pDX, IDC\_EDIT5, m\_rx\_Thread);

}

BEGIN\_MESSAGE\_MAP(CUDPClient\_thdDlg, CDialogEx)

ON\_WM\_SYSCOMMAND()

ON\_WM\_PAINT()

ON\_WM\_QUERYDRAGICON()

ON\_BN\_CLICKED(IDC\_SEND, &CUDPClient\_thdDlg::OnBnClickedSend)

ON\_BN\_CLICKED(IDC\_CLOSE, &CUDPClient\_thdDlg::OnBnClickedClose)

END\_MESSAGE\_MAP()

UINT TXThread(LPVOID arg) //보내는 부분, 따라하기 외 구현한 부분//////////////////////////////

{

ThreadArg\_Send \*pArg = (ThreadArg\_Send \*)arg; //ThreadArg 객체

CStringList \*plist = pArg->pList;

CUDPClient\_thdDlg \*pDlg = (CUDPClient\_thdDlg\*)pArg->pDlg;

CString message;

pDlg->m\_tx\_edit.GetWindowTextW(message);

CString SocketAddress = \_T("127.0.0.1"); //addr

UINT SocketPort = 8000; //Port

CStringList \* stringlist;

CList<Frame> \*alist = pArg->pAckList;

CList<Frame> \*ack\_list;

while (pArg->Thread\_run) {

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

Frame temp;

memset(&temp, 0, sizeof(temp));

current\_pos = pos;

tx\_cs.Lock();

CString str = plist->GetNext(pos);

tx\_cs.Unlock();

stringlist = new CStringList;

stringlist->AddTail(str);

int length;

int length\_w = str.GetLength();

if (str.Find('`') >= 0 || str.GetLength() == 17)

{

str.Remove('`');

temp.end\_game = '`';

}

if (str.Find('/') >= 0)

{

str.Remove('/');

temp.end\_game = '/';

}

CString message;

pDlg->m\_tx\_edit.GetWindowTextW(message);

pDlg->m\_tx\_edit.SetWindowText(message);

wchar\_t\* dataW = new wchar\_t[length\_w];

dataW = (wchar\_t\*)(str.GetString());

length = WideCharToMultiByte(CP\_ACP, 0, dataW, -1, NULL, 0, NULL, NULL);

char \*dataMB = new char[length];

memset(dataMB, 0, sizeof(dataMB));

WideCharToMultiByte(CP\_ACP, 0, dataW, -1, dataMB, length, 0, 0);

/////////////////////////////////

memcpy(temp.data, dataMB, 17);////

////////////////////////////////

temp.sqN = sq++;

temp.flag\_s = SEND;

temp.checkSum = Checksum((unsigned short \*)&temp, sizeof(Frame));

CString tx;

tx.Format(\_T("seqNum:%d, checksum:%d, flag\_s:%c\r\n"), (int)temp.sqN, temp.checkSum, temp.flag\_s);

CString TXmessage;

pDlg->m\_tx\_edit.GetWindowText(TXmessage);

pDlg->m\_pDataSocket->SendToEx(&temp, sizeof(temp), SocketPort, SocketAddress);

int len = pDlg->m\_tx\_Thread.GetWindowTextLengthW();

pDlg->m\_tx\_Thread.SetSel(len, len);

pDlg->m\_tx\_Thread.ReplaceSel(tx);

pDlg->m\_tx\_Thread.SetWindowTextW(tx);

pDlg->m\_tx\_Thread.LineScroll(pDlg->m\_tx\_Thread.GetLineCount());

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount());

Sleep(50);

char status = ACK;

POSITION ack\_Pos = alist->GetHeadPosition();

POSITION ack\_Current\_pos;

bool flag\_t = 0;

//pTime\_th = AfxBeginThread(TimeOutCheckThread, (LPVOID)&flag\_t);

if (flag\_t != 1)//while이랑 토글

{

if (ack\_Pos == NULL)

AfxMessageBox(\_T("액크 리스트 널이다"));

if (ack\_Pos != NULL)

{

ack\_Current\_pos = ack\_Pos;

Frame ack\_temp = alist->GetNext(ack\_Pos);

CString ack\_txt;

ack\_txt.Format(\_T("Receive->ACK Sq : %d flag : %c\r\n"), ack\_temp.sqN, ack\_temp.flag\_s);

pDlg->m\_rx\_Thread.SetWindowTextW(ack\_txt);

status = ack\_temp.flag\_s;

alist->RemoveAt(ack\_Current\_pos);

}

}

if (flag\_t == 1 || status != ACK)

{

temp.flag\_s = RESEND;

tx\_cs.Lock();

pDlg->m\_pDataSocket->SendToEx(&temp, sizeof(temp), SocketPort, SocketAddress);

tx\_cs.Unlock();

}

plist->RemoveAt(current\_pos);

}

Sleep(10);

}

return 0;

}

static CString Buf = \_T("");

UINT RXThread(LPVOID arg)//Receive (Rx)

{

ThreadArg\_Receive \*pArg = (ThreadArg\_Receive \*)arg; //스레드 인자 포인터 pArg생성하여 매개변수 arg를 대입한다.

CList<Frame> \*plist = pArg->pStringList; //리스트 포인터 plist를 생성하여 pArg->pList를 대입한다.

CUDPClient\_thdDlg \*pDlg = (CUDPClient\_thdDlg \*)pArg->pDlg; //dlg포인터 pDlg생성하여 pArg->pList를 대입한다.

UINT port = 8000; CString addr = \_T("127.0.0.1");

CStringList \* stringlist;

while (pArg->Thread\_run)

{

POSITION pos = plist->GetHeadPosition();

POSITION current\_pos;

while (pos != NULL)

{

Frame ack\_frame;

current\_pos = pos;

rx\_cs.Lock(); //Critical Section method Lock()을 다른스레드가 접근 못 하게 함

Frame frame = plist->GetNext(pos); //pos의 다음위치를 plist가 가르키게하고 그 값을 str변수에 저장시켜준다.

rx\_cs.Unlock(); //다른 스레드의 접근을 가능하게 한다.

unsigned short kk = Checksum((unsigned short\*)&frame, sizeof(Frame));

if (kk ==0)//CHECKSUM OK

{

CString temp;

ack\_frame.sqN = frame.sqN + 1;

ack\_frame.flag\_s = ACK;

rx\_cs.Lock();

pDlg->m\_pDataSocket->SendToEx(&ack\_frame, sizeof(Frame), port, addr);

rx\_cs.Unlock();

temp.Format(\_T("seqNum: %d, flag\_s :%c\r\n"), (int)ack\_frame.sqN, ack\_frame.flag\_s);

pDlg->m\_tx\_Thread.SetWindowTextW(temp);

stringlist = new CStringList;

stringlist->AddTail((CString)frame.data);

int len = pDlg->m\_rx\_Thread.GetWindowTextLengthW();

pDlg->m\_rx\_Thread.SetSel(len, len);

temp.Format(\_T("seqNum:%d, checksum:%d, flag\_s:%c, end : %c\r\n"), (int)frame.sqN, frame.checkSum,frame.flag\_s, frame.end\_game);

pDlg->m\_rx\_Thread.ReplaceSel(temp);

CString message;

pDlg->m\_tx\_edit.GetWindowText(message); //dialoge에 접근하여 창에 쓰여진 message를 얻어옴

POSITION fpos = stringlist->GetHeadPosition();

POSITION fcurrent\_pos;

while (fpos != NULL)

{

fcurrent\_pos = fpos;

CString str = stringlist->GetNext(fpos);

//AfxMessageBox(str);

if ((frame.end\_game != '`') && (frame.end\_game != '/') && (str.GetLength() <= 16))

{

str = \_T("서버 : ") + str;

message += str + \_T("\r\n");

message += "\n";

}

if ((frame.end\_game == '`') )

{

if (str.GetLength() == 17)

{

str = str.Mid(0, 16);

}

Buf = Buf + str;

//AfxMessageBox(Buf);

}

if (frame.end\_game == '/')

{

Buf = Buf + str;

//AfxMessageBox(Buf);

str = Buf;

Buf = \_T("");

str.Remove('`'); str.Remove('/');

str = \_T("서버 : ") + str;

message += str + \_T("\r\n");

message += "\n";

}

stringlist->RemoveAt(fcurrent\_pos);

}

pDlg->m\_tx\_edit.SetWindowText(message); //메세지를 창에 출력한다

pDlg->m\_tx\_edit.LineScroll(pDlg->m\_tx\_edit.GetLineCount()); //텍스트 스크롤

plist->RemoveAt(current\_pos); //리스트에서 꺼낸 데이터 삭제

}

else

{

ack\_frame.sqN = frame.sqN+1;

ack\_frame.flag\_s = NACK;

rx\_cs.Lock();

pDlg->m\_pDataSocket->SendToEx(&ack\_frame, sizeof(Frame), port, addr);

break;

rx\_cs.Unlock();

}

}

Sleep(10);

}

return 0;

}

// CUDPClient\_thdDlg 메시지 처리기

BOOL CUDPClient\_thdDlg::OnInitDialog()

{

CDialogEx::OnInitDialog();

// 시스템 메뉴에 "정보..." 메뉴 항목을 추가합니다.

// IDM\_ABOUTBOX는 시스템 명령 범위에 있어야 합니다.

ASSERT((IDM\_ABOUTBOX & 0xFFF0) == IDM\_ABOUTBOX);

ASSERT(IDM\_ABOUTBOX < 0xF000);

CMenu\* pSysMenu = GetSystemMenu(FALSE);

if (pSysMenu != NULL)

{

BOOL bNameValid;

CString strAboutMenu;

bNameValid = strAboutMenu.LoadString(IDS\_ABOUTBOX);

ASSERT(bNameValid);

if (!strAboutMenu.IsEmpty())

{

pSysMenu->AppendMenu(MF\_SEPARATOR);

pSysMenu->AppendMenu(MF\_STRING, IDM\_ABOUTBOX, strAboutMenu);

}

}

// 이 대화 상자의 아이콘을 설정합니다. 응용 프로그램의 주 창이 대화 상자가 아닐 경우에는

// 프레임워크가 이 작업을 자동으로 수행합니다.

SetIcon(m\_hIcon, TRUE); // 큰 아이콘을 설정합니다.

SetIcon(m\_hIcon, FALSE); // 작은 아이콘을 설정합니다.

// TODO: 여기에 추가 초기화 작업을 추가합니다.

//arg1은 trans다

CStringList\* strList = new CStringList;

arg1.pList = strList;

arg1.Thread\_run = 1;

arg1.pDlg = this;

CList<Frame>\* ackList1 = new CList<Frame>;

arg1.pAckList = ackList1;

CList<Frame>\* frameList = new CList<Frame>;

arg2.pStringList = frameList;

arg2.Thread\_run = 1;

arg2.pDlg = this;

CList<Frame>\* ackList2 = new CList<Frame>;

arg2.pAckList = ackList2;

m\_pDataSocket = NULL;

m\_ipaddr.SetWindowTextW(\_T("127.0.0.1"));

pThread1 = AfxBeginThread(TXThread, (LPVOID)&arg1);

pThread2 = AfxBeginThread(RXThread, (LPVOID)&arg2);

if (m\_pDataSocket == NULL) {

m\_pDataSocket = new CDataSocket(this);

CString addr;

m\_ipaddr.GetWindowText(addr);

if (m\_pDataSocket->Create(8001, SOCK\_DGRAM)) {

m\_tx\_edit\_short.SetFocus();

}

else {

delete m\_pDataSocket;

m\_pDataSocket = NULL;

}

}

else {

//MessageBox(\_T("서버에 이미 접속됨!"), \_T("알림"), MB\_ICONINFORMATION);

m\_tx\_edit\_short.SetFocus();

}

return TRUE; // 포커스를 컨트롤에 설정하지 않으면 TRUE를 반환합니다.

}

void CUDPClient\_thdDlg::OnSysCommand(UINT nID, LPARAM lParam)

{

if ((nID & 0xFFF0) == IDM\_ABOUTBOX)

{

CAboutDlg dlgAbout;

dlgAbout.DoModal();

}

else

{

CDialogEx::OnSysCommand(nID, lParam);

}

}

// 대화 상자에 최소화 단추를 추가할 경우 아이콘을 그리려면

// 아래 코드가 필요합니다. 문서/뷰 모델을 사용하는 MFC 응용 프로그램의 경우에는

// 프레임워크에서 이 작업을 자동으로 수행합니다.

void CUDPClient\_thdDlg::OnPaint()

{

if (IsIconic())

{

CPaintDC dc(this); // 그리기를 위한 디바이스 컨텍스트입니다.

SendMessage(WM\_ICONERASEBKGND, reinterpret\_cast<WPARAM>(dc.GetSafeHdc()), 0);

// 클라이언트 사각형에서 아이콘을 가운데에 맞춥니다.

int cxIcon = GetSystemMetrics(SM\_CXICON);

int cyIcon = GetSystemMetrics(SM\_CYICON);

CRect rect;

GetClientRect(&rect);

int x = (rect.Width() - cxIcon + 1) / 2;

int y = (rect.Height() - cyIcon + 1) / 2;

// 아이콘을 그립니다.

dc.DrawIcon(x, y, m\_hIcon);

}

else

{

CDialogEx::OnPaint();

}

}

// 사용자가 최소화된 창을 끄는 동안에 커서가 표시되도록 시스템에서

// 이 함수를 호출합니다.

HCURSOR CUDPClient\_thdDlg::OnQueryDragIcon()

{

return static\_cast<HCURSOR>(m\_hIcon);

}

void CUDPClient\_thdDlg::OnBnClickedSend()

{

CString tx\_Smessage = NULL;

m\_tx\_edit\_short.GetWindowText(tx\_Smessage);

int length = 0;

CString \*temp;

length = tx\_Smessage.GetLength();

if ((length <= 16)) { //만약에 문자열길이가 16바이트이거나

//또는 16바이트 보다 작으면 길이를 1로 주어지게.

CString \*temp = new CString[1];

temp[0] = tx\_Smessage.Mid(0, 16);

tx\_cs.Lock();

arg1.pList->AddTail(temp[0]);

tx\_cs.Unlock();

}

else if ((length > 16) && (length % 16) != 0) { //만약 문자열 길이가 16보다 클 경우 길이가

CString \*temp = new CString[(length / 16) + 1];

for (int i = 0; i < (length / 16) + 1; i++)

{

if (i == (length / 16))

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("/");

else

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("`");

}

tx\_cs.Lock();

for (int i = 0; i < (length / 16) + 1; i++) {

arg1.pList->AddTail(temp[i]);

}

tx\_cs.Unlock();

}

else {

CString \*temp = new CString[(length / 16) + 1];

for (int i = 0; i < (length / 16) + 1; i++)

{

if (i == (length / 16))

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("/");

else

temp[i] = tx\_Smessage.Mid(i \* 16, i \* 16 + 16) + \_T("`");

}

tx\_cs.Lock();

for (int i = 0; i < (length / 16) + 1; i++) {

arg1.pList->AddTail(temp[i]);

}

tx\_cs.Unlock();

}

m\_tx\_edit\_short.SetWindowText(\_T(""));//m\_tx\_edit\_short 비우기

m\_tx\_edit\_short.SetFocus();//입력 상태로 만들기

tx\_Smessage = \_T("클라 : ") + tx\_Smessage;

tx\_Smessage += "\r\n";

int len = m\_tx\_edit.GetWindowTextLengthW();

m\_tx\_edit.SetSel(len, len);

m\_tx\_edit.ReplaceSel(tx\_Smessage);//보낸 tx\_Smessage 띄우기

}

void CUDPClient\_thdDlg::OnBnClickedClose()

{

// TODO: 여기에 컨트롤 알림 처리기 코드를 추가합니다.

if (m\_pDataSocket == NULL) //m\_pDataSocket 이 NULL상태라면

{

MessageBox(\_T("서버에 접속 안함!"), \_T("알림"), MB\_ICONINFORMATION);

}

else

{

arg1.Thread\_run = 0; //To disconnect set Thread\_run = 1

arg2.Thread\_run = 0; //To disconnect set Thread\_run = 1

m\_pDataSocket->Close();

delete m\_pDataSocket;

m\_pDataSocket = NULL;

int len = m\_tx\_edit.GetWindowTextLengthW();

CString message = \_T("\n### 접속 종료 ###\r\n\r\n");

m\_tx\_edit.SetSel(len, len); //Set Cursor place

m\_tx\_edit.ReplaceSel(message);

}

}

void CUDPClient\_thdDlg::ProcessReceive(CDataSocket\* pSocket, int nErrorCode)

{

Frame get;

Frame\* Data = &get;

int nbytes;

CString SocketAddress = \_T("127.0.0.1");

UINT SocketPort = 8000;

nbytes = pSocket->ReceiveFromEx(&get, sizeof(Frame), SocketAddress, SocketPort, 0);

if (get.flag\_s == ACK || get.flag\_s == NACK)

{

tx\_cs.Lock();

arg1.pAckList->AddTail(\*Data);

CString temp;

temp.Format(\_T("AckFrame's Seq.No : %d, Flag\_s : %c"), get.sqN, get.flag\_s);

AfxMessageBox(temp);

tx\_cs.Unlock();

}

else if (get.sqN == SEND || get.flag\_s == RESEND)

{

rx\_cs.Lock();

arg2.pStringList->AddTail(\*Data);

rx\_cs.Unlock();

}

else {

rx\_cs.Lock();

arg2.pStringList->AddTail(\*Data);

arg2.pAckList->AddTail(\*Data);

rx\_cs.Unlock();

}

}

BOOL CUDPClient\_thdDlg::PreTranslateMessage(MSG\* pMsg)

{

// TODO: 여기에 특수화된 코드를 추가 및/또는 기본 클래스를 호출합니다.

return CDialogEx::PreTranslateMessage(pMsg);

if (pMsg->message == WM\_KEYDOWN)

{

if (pMsg->wParam == VK\_RETURN)

{

OnBnClickedSend();

}

else if (pMsg->wParam == VK\_ESCAPE)

{

OnOK();

}

}

}

unsigned short Checksum(unsigned short \*ptr, int size)

{

Frame \*frame = (Frame \*)ptr; //checksum을 생성할 문자

unsigned short checksum = 0;

unsigned short oddbyte;

register short result;

while (size > 1)

{

checksum += \*ptr++;

size -= 2;

}

if (size == 1)

{

oddbyte = 0;

\*((u\_char\*)&oddbyte) = \*((u\_char\*)ptr);//oddbyte가 가리키는 주소에 들어있는 거에 ptr이 가리키는 내용을 넣음

checksum += oddbyte;

}

checksum = (checksum >> 16) + checksum; //carry처리

checksum = ~checksum; //compliment

return checksum;

}

UINT TimeOutCheckThread(LPVOID arg)

{

int \*flag\_t = (int \*)arg;

Sleep(5000);

(\*flag\_t) = 1 - (\*flag\_t);

//\*(bool\*)arg = false;

return 0;

}