Міністерство освіти і науки України

Центральноукраїнський національний технічний університет

**МЕХАНІКО-ТЕХНОЛОГІЧНИЙ ФАКУЛЬТЕТ**

Кафедра кібербезпеки та програмного забезпечення

**Звіт**

з виконаної лабораторної роботи № 1

дисципліни “ Архітектура комп’ютерів”

Виконав :

студент академічної групи КІ-15

Аннаєв А. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірив :

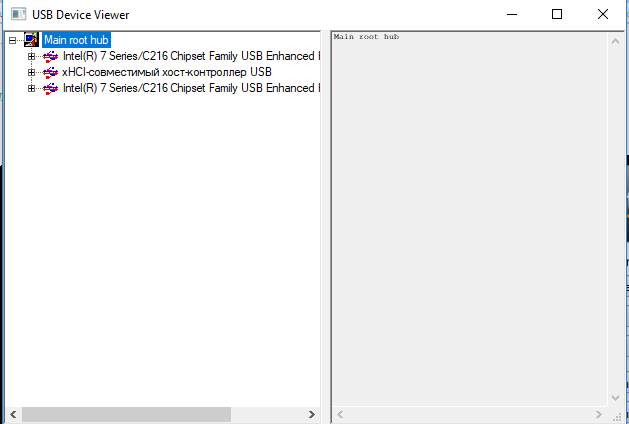
Викладач

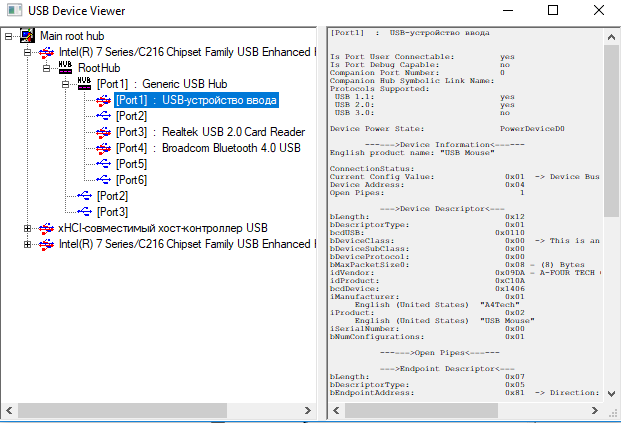
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Кропивницкий- 2018

Лабораторная работа № 1

Тема: Клиентское программное обеспечения для мониторинга USB устройств, используя набор инструментов WDK





//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

int WINAPI WinMain(\_In\_ HINSTANCE hInstance, \_In\_opt\_ HINSTANCE hPrevInstance, \_In\_ LPSTR lpszCmdLine, \_In\_ int nCmdShow);

BOOL CreateMainWindow(int nCmdShow);

VOID ResizeWindows(BOOL bSizeBar, int BarLocation);

LRESULT CALLBACK MainDlgProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

VOID DestroyTree(VOID);

VOID RefreshTree(VOID);

VOID WalkTree(\_In\_ HTREEITEM hTreeItem, \_In\_ LPFNTREECALLBACK lpfnTreeCallback, \_In\_opt\_ PVOID pContext);

VOID ExpandItem(HWND hTreeWnd, HTREEITEM hTreeItem, PVOID pContext);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

BOOL USBView\_OnInitDialog(HWND hWnd, HWND hWndFocus, LPARAM lParam);

VOID USBView\_OnClose(HWND hWnd);

VOID USBView\_OnLButtonDown(HWND hWnd, BOOL fDoubleClick, int x, int y, UINT keyFlags);

VOID USBView\_OnLButtonUp(HWND hWnd, int x, int y, UINT keyFlags);

VOID USBView\_OnSize(HWND hWnd, UINT state, int cx, int cy);

LRESULT USBView\_OnNotify(HWND hWnd, int DlgItem, LPNMHDR lpNMHdr);

BOOL USBView\_OnDeviceChange(HWND hwnd, UINT uEvent, DWORD dwEventData);

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

WinMain()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

int WINAPI WinMain(\_In\_ HINSTANCE hInstance, \_In\_opt\_ HINSTANCE hPrevInstance, \_In\_ LPSTR lpszCmdLine, \_In\_ int nCmdShow) {

MSG msg;

HACCEL hAccel;

int retStatus = 0;

ghInstance = hInstance;

ghSplitCursor = LoadCursor(ghInstance, MAKEINTRESOURCE(IDC\_SPLIT));

hAccel = LoadAccelerators(ghInstance, MAKEINTRESOURCE(IDACCEL));

if (!CreateTextBuffer()) {

return retStatus;

}

if (CreateMainWindow(nCmdShow)) {

while (GetMessage(&msg, NULL, 0, 0)) {

if (!TranslateAccelerator(ghMainWnd, hAccel, &msg) && !IsDialogMessage(ghMainWnd, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

retStatus = 1;

}

DestroyTextBuffer();

CHECKFORLEAKS();

return retStatus;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

CreateMainWindow()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BOOL CreateMainWindow(int nCmdShow) {

RECT rc;

InitCommonControls();

ghMainWnd = CreateDialog(ghInstance, MAKEINTRESOURCE(IDD\_MAINDIALOG), NULL, (DLGPROC)MainDlgProc);

GetWindowRect(ghMainWnd, &rc);

gBarLocation = (rc.right - rc.left) / 2;

ResizeWindows(FALSE, 0);

ShowWindow(ghMainWnd, nCmdShow);

UpdateWindow(ghMainWnd);

return TRUE;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ResizeWindows()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID ResizeWindows(BOOL bSizeBar, int BarLocation)

{

RECT MainClientRect;

RECT MainWindowRect;

RECT TreeWindowRect;

int right;

if (!bSizeBar)

BarLocation = gBarLocation;

GetClientRect(ghMainWnd, &MainClientRect);

if (bSizeBar) {

if (BarLocation <

GetSystemMetrics(SM\_CXSCREEN) / WINDOWSCALEFACTOR)

{

return;

}

if ((MainClientRect.right - BarLocation) <

GetSystemMetrics(SM\_CXSCREEN) / WINDOWSCALEFACTOR)

{

return;

}

}

gBarLocation = BarLocation;

MoveWindow(ghTreeWnd, 0, 0, BarLocation, MainClientRect.bottom , TRUE);

GetWindowRect(ghTreeWnd, &TreeWindowRect);

GetWindowRect(ghMainWnd, &MainWindowRect);

right = TreeWindowRect.right - MainWindowRect.left;

MoveWindow(ghEditWnd, right + SIZEBAR, 0, MainClientRect.right - (right + SIZEBAR), MainClientRect.bottom, TRUE);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

MainWndProc()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LRESULT CALLBACK MainDlgProc(HWND hWnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {

switch (uMsg)

{

HANDLE\_MSG(hWnd, WM\_INITDIALOG, USBView\_OnInitDialog);

HANDLE\_MSG(hWnd, WM\_CLOSE, USBView\_OnClose);

HANDLE\_MSG(hWnd, WM\_LBUTTONDOWN, USBView\_OnLButtonDown);

HANDLE\_MSG(hWnd, WM\_LBUTTONUP, USBView\_OnLButtonUp);

HANDLE\_MSG(hWnd, WM\_SIZE, USBView\_OnSize);

HANDLE\_MSG(hWnd, WM\_NOTIFY, USBView\_OnNotify);

HANDLE\_MSG(hWnd, WM\_DEVICECHANGE, USBView\_OnDeviceChange);

}

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnInitDialog()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BOOL USBView\_OnInitDialog(HWND hWnd, HWND hWndFocus, LPARAM lParam)

{

HFONT hFont;

HIMAGELIST himl;

HICON hicon;

DEV\_BROADCAST\_DEVICEINTERFACE broadcastInterface;

// Register to receive notification when a USB device is plugged in.

broadcastInterface.dbcc\_size = sizeof(DEV\_BROADCAST\_DEVICEINTERFACE);

broadcastInterface.dbcc\_devicetype = DBT\_DEVTYP\_DEVICEINTERFACE;

memcpy(&(broadcastInterface.dbcc\_classguid), &(GUID\_DEVINTERFACE\_USB\_DEVICE), sizeof(struct \_GUID));

gNotifyDevHandle = RegisterDeviceNotification(hWnd, &broadcastInterface, DEVICE\_NOTIFY\_WINDOW\_HANDLE);

memcpy(&(broadcastInterface.dbcc\_classguid), &(GUID\_CLASS\_USBHUB), sizeof(struct \_GUID));

gNotifyHubHandle = RegisterDeviceNotification(hWnd, &broadcastInterface, DEVICE\_NOTIFY\_WINDOW\_HANDLE);

gHubList.DeviceInfo = INVALID\_HANDLE\_VALUE;

InitializeListHead(&gHubList.ListHead);

gDeviceList.DeviceInfo = INVALID\_HANDLE\_VALUE;

InitializeListHead(&gDeviceList.ListHead);

ghTreeWnd = GetDlgItem(hWnd, IDC\_TREE);

himl = ImageList\_Create(15, 15, FALSE, 2, 0);

if (himl != NULL)

{

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_ICON));

giGoodDevice = ImageList\_AddIcon(himl, hicon);

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_BADICON));

giBadDevice = ImageList\_AddIcon(himl, hicon);

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_COMPUTER));

giComputer = ImageList\_AddIcon(himl, hicon);

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_HUB));

giHub = ImageList\_AddIcon(himl, hicon);

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_NODEVICE));

giNoDevice = ImageList\_AddIcon(himl, hicon);

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_SSICON));

giGoodSsDevice = ImageList\_AddIcon(himl, hicon);

hicon = LoadIcon(ghInstance, MAKEINTRESOURCE(IDI\_NOSSDEVICE));

giNoSsDevice = ImageList\_AddIcon(himl, hicon);

TreeView\_SetImageList(ghTreeWnd, himl, TVSIL\_NORMAL);

}

ghEditWnd = GetDlgItem(hWnd, IDC\_EDIT);

#ifdef H264\_SUPPORT

// set the edit control to have a max text limit size

SendMessage(ghEditWnd, EM\_LIMITTEXT, 0 /\* USE DEFAULT MAX\*/, 0);

#endif

{

CHAR pszFont[256];

CHAR pszHeight[8];

memset(pszFont, 0, sizeof(pszFont));

LoadString(ghInstance, IDS\_STANDARD\_FONT, pszFont, sizeof(pszFont) - 1);

memset(pszHeight, 0, sizeof(pszHeight));

LoadString(ghInstance, IDS\_STANDARD\_FONT\_HEIGHT, pszHeight, sizeof(pszHeight) - 1);

hFont = CreateFont((int)pszHeight[0], 0, 0, 0,

400, 0, 0, 0,

0, 1, 2, 1,

49, pszFont);

}

SendMessage(ghEditWnd, WM\_SETFONT, (WPARAM)hFont, 0);

RefreshTree();

return FALSE;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnClose()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID USBView\_OnClose(HWND hWnd)

{

UNREFERENCED\_PARAMETER(hWnd);

DestroyTree();

PostQuitMessage(0);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnLButtonDown()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID USBView\_OnLButtonDown(HWND hWnd, BOOL fDoubleClick, int x, int y, UINT keyFlags)

{

gbButtonDown = TRUE;

SetCapture(hWnd);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnLButtonUp()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID USBView\_OnLButtonUp(HWND hWnd, int x, int y, UINT keyFlags)

{

gbButtonDown = FALSE;

ReleaseCapture();

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnSize();

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID USBView\_OnSize(HWND hWnd, UINT state, int cx, int cy)

{

ResizeWindows(FALSE, 0);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnNotify()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

LRESULT USBView\_OnNotify(HWND hWnd, int DlgItem, LPNMHDR lpNMHdr)

{

if (lpNMHdr->code == TVN\_SELCHANGED)

{

HTREEITEM hTreeItem;

hTreeItem = ((NM\_TREEVIEW \*)lpNMHdr)->itemNew.hItem;

if (hTreeItem)

UpdateEditControl(ghEditWnd, ghTreeWnd, hTreeItem);

}

return 0;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

USBView\_OnDeviceChange()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

BOOL USBView\_OnDeviceChange(HWND hwnd, UINT uEvent, DWORD dwEventData)

{

if (gDoAutoRefresh)

{

switch (uEvent)

{

case DBT\_DEVICEARRIVAL:

case DBT\_DEVICEREMOVECOMPLETE:

RefreshTree();

break;

}

}

return TRUE;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

DestroyTree()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID DestroyTree(VOID)

{

TreeView\_SelectItem(ghTreeWnd, NULL);

if (ghTreeRoot)

{

WalkTree(ghTreeRoot, CleanupItem, NULL);

TreeView\_DeleteAllItems(ghTreeWnd);

ghTreeRoot = NULL;

}

ClearDeviceList(&gDeviceList);

ClearDeviceList(&gHubList);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

RefreshTree()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID RefreshTree(VOID)

{

CHAR statusText[128];

ULONG devicesConnected;

SetWindowText(ghEditWnd, "");

DestroyTree();

ghTreeRoot = AddLeaf(TVI\_ROOT, 0, "Main root hub", ComputerIcon);

if (ghTreeRoot != NULL)

{

EnumerateHostControllers(ghTreeRoot, &devicesConnected);

WalkTree(ghTreeRoot, ExpandItem, NULL);

memset(statusText, 0, sizeof(statusText));

StringCchPrintf(statusText, sizeof(statusText),

#ifdef H264\_SUPPORT

"UVC Spec Version: %d.%d Version: %d.%d Devices Connected: %d Hubs Connected: %d",

UVC\_SPEC\_MAJOR\_VERSION, UVC\_SPEC\_MINOR\_VERSION, USBVIEW\_MAJOR\_VERSION, USBVIEW\_MINOR\_VERSION,

devicesConnected, TotalHubs);

#else

"Devices Connected: %d Hubs Connected: %d",

devicesConnected, TotalHubs);

#endif

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

AddLeaf()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

HTREEITEM AddLeaf(HTREEITEM hTreeParent, LPARAM lParam, \_In\_ LPTSTR lpszText, TREEICON TreeIcon)

{

TV\_INSERTSTRUCT tvins;

HTREEITEM hti;

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

tvins.hParent = hTreeParent;

tvins.hInsertAfter = TVI\_LAST;

tvins.item.mask = TVIF\_TEXT | TVIF\_PARAM;

tvins.item.pszText = lpszText;

tvins.item.lParam = lParam;

hti = TreeView\_InsertItem(ghTreeWnd, &tvins);

tvins.item.mask = TVIF\_IMAGE | TVIF\_SELECTEDIMAGE;

tvins.item.hItem = hti;

switch (TreeIcon)

{

case ComputerIcon:

tvins.item.iImage = giComputer;

tvins.item.iSelectedImage = giComputer;

break;

case HubIcon:

tvins.item.iImage = giHub;

tvins.item.iSelectedImage = giHub;

break;

case NoDeviceIcon:

tvins.item.iImage = giNoDevice;

tvins.item.iSelectedImage = giNoDevice;

break;

case GoodDeviceIcon:

tvins.item.iImage = giGoodDevice;

tvins.item.iSelectedImage = giGoodDevice;

break;

case GoodSsDeviceIcon:

tvins.item.iImage = giGoodSsDevice;

tvins.item.iSelectedImage = giGoodSsDevice;

break;

case NoSsDeviceIcon:

tvins.item.iImage = giNoSsDevice;

tvins.item.iSelectedImage = giNoSsDevice;

break;

case BadDeviceIcon:

default:

tvins.item.iImage = giBadDevice;

tvins.item.iSelectedImage = giBadDevice;

break;

}

TreeView\_SetItem(ghTreeWnd, &tvins.item);

return hti;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

WalkTreeTopDown()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID WalkTreeTopDown(\_In\_ HTREEITEM hTreeItem, \_In\_ LPFNTREECALLBACK lpfnTreeCallback, \_In\_opt\_ PVOID pContext, \_In\_opt\_ LPFNTREENOTIFYCALLBACK lpfnTreeNotifyCallback)

{

if (hTreeItem)

{

HTREEITEM hTreeChild = TreeView\_GetChild(ghTreeWnd, hTreeItem);

HTREEITEM hTreeSibling = TreeView\_GetNextSibling(ghTreeWnd, hTreeItem);

(\*lpfnTreeCallback)(ghTreeWnd, hTreeItem, pContext);

if (hTreeChild)

WalkTreeTopDown(hTreeChild, lpfnTreeCallback, pContext, lpfnTreeNotifyCallback);

if (hTreeSibling)

WalkTreeTopDown(hTreeSibling, lpfnTreeCallback, pContext, lpfnTreeNotifyCallback);

else if (lpfnTreeNotifyCallback != NULL)

(\*lpfnTreeNotifyCallback)(pContext);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

WalkTree()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID WalkTree(\_In\_ HTREEITEM hTreeItem, \_In\_ LPFNTREECALLBACK lpfnTreeCallback, \_In\_opt\_ PVOID pContext) {

if (hTreeItem) {

WalkTree(TreeView\_GetChild(ghTreeWnd, hTreeItem), lpfnTreeCallback, pContext);

(\*lpfnTreeCallback)(ghTreeWnd, hTreeItem, pContext);

WalkTree(TreeView\_GetNextSibling(ghTreeWnd, hTreeItem), lpfnTreeCallback, pContext);

}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ExpandItem()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

VOID ExpandItem(HWND hTreeWnd, HTREEITEM hTreeItem, PVOID pContext) {

UNREFERENCED\_PARAMETER(pContext);

TreeView\_Expand(hTreeWnd, hTreeItem, TVE\_EXPAND);

}