
Subject: Windows Service?

Posted by [WebChaot](#) on Mon, 04 Dec 2006 10:33:51 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hi all!

I know, a windows service is platform specific and therefore not in this forum (until now).

But does anybody know, if it is (or should be) possible to create a windows service with ultimate++ - and how to start?

We want to use U++ for all our new (or updated) applications in future - so now we have to re-develop an existing service application - and we want to do it with U++.

I would be happy, if someone can point me to a piece of code or other useful information about this topic.

Thank you in advance,

WebChaot.

Subject: Re: Windows Service?

Posted by [WebChaot](#) on Mon, 04 Dec 2006 11:56:49 GMT

[View Forum Message](#) <> [Reply to Message](#)

We found a solution! If someone is interested, here is the code:

```
#include <Core/Core.h>
```

```
/*  
=====
```

```
WinNT service example. Based on code found in internet. Here is the original header:
```

```
"You may use Makc <makc.the.great@gmail.com> for credits ;)"
```

```
"Windows service C++ implementation based on MSDN sample by Nigel Thompson (Microsoft  
Developer Network Technology Group) November 1995"
```

```
=====
```

```
===== */  
  
#include <windows.h>  
#include <winsock.h>  
#define snprintf _snprintf  
#pragma comment(lib,"ws2_32")
```

```

#pragma comment(lib,"advapi32")

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

// windows service class - must be named as "CNTService"! Dont know why - but when you
// rename the class, you can not stop the service (?).

class CNTService
{
public:
    bool Init();
    void Run();

    CNTService(const char* szServiceName);
    ~CNTService();

    bool IsInstalled();
    bool Install();
    bool Uninstall();

    bool StartService();
    void SetStatus(DWORD dwState);
    bool Initialize();

    static void WINAPI ServiceMain(DWORD dwArgc, LPTSTR* lpszArgv);
    static void WINAPI Handler(DWORD dwOpcode);

    char m_szServiceName[64];
    int m_iMajorVersion;
    int m_iMinorVersion;
    SERVICE_STATUS_HANDLE m_hServiceStatus;
    SERVICE_STATUS m_Status;

    bool m_bIsRunning;

    static CNTService* m_pThis; // this code is older than HandlerEx :(
private:
    HANDLE m_hEventSource;
};

void CNTService::Run()
{
    /* Serve connections until SERVICE_CONTROL_STOP received */
    while (m_bIsRunning)

```

```

{
    // do something
    Sleep(500);
}
}

```

```

bool CNTService::Init()
{
    // do something to init

    return true;
}

```

```

CNTService* CNTService::m_pThis = NULL;

```

```

CNTService::CNTService(const char* szServiceName)
{
    m_pThis = this;

    // set default service name
    strncpy(m_szServiceName, szServiceName, sizeof(m_szServiceName)-1);
    m_hEventSource = NULL;

    // set initial service status
    m_hServiceStatus = NULL;
    m_Status.dwServiceType = SERVICE_WIN32_OWN_PROCESS;
    m_Status.dwCurrentState = SERVICE_STOPPED;
    m_Status.dwControlsAccepted = SERVICE_ACCEPT_STOP |
SERVICE_ACCEPT_SHUTDOWN;
    m_Status.dwWin32ExitCode = 0;
    m_Status.dwServiceSpecificExitCode = 0;
    m_Status.dwCheckPoint = 0;
    m_Status.dwWaitHint = 0;
    m_bIsRunning = FALSE;
}

```

```

CNTService::~~CNTService()
{
    if (m_hEventSource)
    {
        ::DeregisterEventSource(m_hEventSource);
    }
}

```

```

bool CNTService::IsInstalled()
{
    bool bResult = FALSE;

    SC_HANDLE hSCM = ::OpenSCManager(NULL,    // local machine
        NULL,    // ServicesActive database
        SC_MANAGER_ALL_ACCESS); // full access

    if (hSCM)
    {
        SC_HANDLE hService = ::OpenService(hSCM,
            m_szServiceName,
            SERVICE_QUERY_CONFIG);
        if (hService)
        {
            bResult = TRUE;
            ::CloseServiceHandle(hService);
        }

        ::CloseServiceHandle(hSCM);
    }
    return bResult;
}

```

```

bool CNTService::Install()
{
    SC_HANDLE hSCM = ::OpenSCManager(NULL,    // local machine
        NULL,    // ServicesActive database
        SC_MANAGER_ALL_ACCESS); // full access

    if (!hSCM) return false;

    char szFilePath[FILENAME_MAX];

    ::GetModuleFileName(NULL, szFilePath, sizeof(szFilePath));

    SC_HANDLE hService = ::CreateService(hSCM,
        m_szServiceName,
        m_szServiceName,
        SERVICE_ALL_ACCESS,
        SERVICE_WIN32_OWN_PROCESS,
        SERVICE_AUTO_START, // start condition
        SERVICE_ERROR_NORMAL,
        szFilePath,
        NULL,
        NULL,
        NULL,

```

```

        NULL,
        NULL);

if (!hService)
{
    ::CloseServiceHandle(hSCM);

    return false;
}

char szKey[256]; HKEY hKey = NULL;

strcpy(szKey, "SYSTEM\\CurrentControlSet\\Services\\EventLog\\Application\\");

strcat(szKey, m_szServiceName);

if (::RegCreateKey(HKEY_LOCAL_MACHINE, szKey, &hKey) != ERROR_SUCCESS)
{
    ::CloseServiceHandle(hService);
    ::CloseServiceHandle(hSCM);

    return false;
}

::RegSetValueEx(hKey,
    "EventMessageFile",
    0,
    REG_EXPAND_SZ,
    (CONST BYTE*)szFilePath,
    (int) strlen(szFilePath) + 1);

DWORD dwData = EVENTLOG_ERROR_TYPE | EVENTLOG_WARNING_TYPE |
EVENTLOG_INFORMATION_TYPE;

::RegSetValueEx(hKey,
    "TypesSupported",
    0,
    REG_DWORD,
    (CONST BYTE*)&dwData,
    sizeof(DWORD));

::RegCloseKey(hKey);

::CloseServiceHandle(hService);
::CloseServiceHandle(hSCM);

return true;
}

```

```

bool CNTService::Uninstall()
{
    SC_HANDLE hSCM = ::OpenSCManager(NULL,    // local machine
        NULL,    // ServicesActive database
        SC_MANAGER_ALL_ACCESS); // full access
    if (!hSCM) return false;

    bool bResult = false;

    SC_HANDLE hService = ::OpenService(hSCM,
        m_szServiceName,
        DELETE);

    if (hService)
    {
        if (::DeleteService(hService))
        {
            bResult = true;
        }

        ::CloseServiceHandle(hService);
    }
    ::CloseServiceHandle(hSCM);

    return bResult;
}

```

```

bool CNTService::StartService()
{
    SERVICE_TABLE_ENTRY st[] =
    {
        {m_szServiceName, ServiceMain},
        {NULL, NULL}
    };

    return ::StartServiceCtrlDispatcher(st);
}

```

```

void CNTService::SetStatus(DWORD dwState)
{
    m_Status.dwCurrentState = dwState;

    ::SetServiceStatus(m_hServiceStatus, &m_Status);
}

```

```

bool CNTService::Initialize()
{
    SetStatus(SERVICE_START_PENDING);

    bool bResult = Init();    // actual initialization

    m_Status.dwWin32ExitCode = GetLastError();
    m_Status.dwCheckPoint = 0;
    m_Status.dwWaitHint = 0;

    if (!bResult)
    {
        SetStatus(SERVICE_STOPPED);

        return false;
    }

    SetStatus(SERVICE_RUNNING);

    return true;
}

void CNTService::ServiceMain(DWORD dwArgc, LPTSTR* lpszArgv)
{
    CNTService* pService = m_pThis;

    pService->m_Status.dwCurrentState = SERVICE_START_PENDING;
    pService->m_hServiceStatus = RegisterServiceCtrlHandler(pService->m_szServiceName,
    Handler);

    if (pService->m_hServiceStatus != NULL)
    {
        if (pService->Initialize())
        {
            pService->m_bIsRunning = true;
            pService->m_Status.dwWin32ExitCode = 0;
            pService->m_Status.dwCheckPoint = 0;
            pService->m_Status.dwWaitHint = 0;
            pService->Run();
        }
        pService->SetStatus(SERVICE_STOPPED);
    }
}

```

```

void CNTService::Handler(DWORD dwOpcode)
{
    CNTService* pService = m_pThis;

    if ((dwOpcode == SERVICE_CONTROL_STOP) ||
        (dwOpcode == SERVICE_CONTROL_SHUTDOWN))
    {
        pService->SetStatus(SERVICE_STOP_PENDING);
        pService->m_bIsRunning = false;    // SERVICE_STOPPED set when Run returns in
        ServiceMain (above)
    }

    ::SetServiceStatus(pService->m_hServiceStatus, &pService->m_Status);
}

```

```

int main(int argc, char* argv[])
{
    int iCode = 1;
    bool bRun = false;

    // Enter here the service name.
    CNTService ntService ("ecdss");

    if (ntService.IsInstalled())
    {
        // uninstall or run
        bRun = true;
        if (argc > 1)
        {
            if (_stricmp(argv[1], "-u") == 0)
            {
                bRun = false;
                if (ntService.Uninstall())
                {
                    iCode = 0;
                }
            }
        }
    }
    else
    {
        // install before run
        if (ntService.Install())
        {
            bRun = true;
        }
    }
}

```



```
if (bRun)
{
    if (ntService.StartService())
    {
        iCode = 0;
    }
    else if (GetLastError() == ERROR_FAILED_SERVICE_CONTROLLER_CONNECT)
    {
        printf("Service installed; to start it, type this command: net start %s\n\n",
ntService.m_szServiceName );
        iCode = 0;
    }
}

return iCode;
}
```

WebChaot.
