Subject: Self-install / uninstall in Win32
Posted by mirek on Wed, 27 Jun 2007 11:16:15 GMT
View Forum Message <> Reply to Message

This is not matured enough to be put into "uppsrc", but I think it is an interesting piece of code anyway: It allows an application .exe to "selfinstall" to Program Files and Desktop, with registration to windows "Remove programs" dialog (and uninstallation):

```
#include "zaci.h"

#define Ptr Ptr_
#define byte byte_
#define CY win32_CY_


#include <winnls.h>
#include <winnetwk.h>

#include <wincon.h>
#include <shlobj.h>

#undef Ptr
#undef byte
#undef CY

String GetShellFolder(const char *name, HKEY type)
{
 return GetWinRegString(name, "Software\\Microsoft\\Windows\\CurrentVersion\\Explorer\\Shell
Folders", type);
}

void DelKey(const char *dir, const char *key)
{
 HKEY hkey;
 if(RegOpenKeyEx(HKEY_LOCAL_MACHINE, dir, 0, KEY_READ, &hkey) != ERROR_SUCCESS)
  return;
 RegDeleteKey(hkey, key);
 RegCloseKey(hkey);
}

void Uninstall(String folder, String name)
{
 String path = GetExeFilePath();
 String bat = AppendFileName(GetShellFolder("Desktop", HKEY_CURRENT_USER),
"removepp.bat");
 String dir = GetFileFolder(path);
```

```
  return;
 DeleteFolderDeep(dir);

 DeleteFile(AppendFileName(GetShellFolder("Desktop", HKEY_CURRENT_USER), name +
".lnk"));

 DelKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall\\" + folder, "DisplayName");
 DelKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall\\" + folder,
"UninstallString");
 DelKey("SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall", folder);

 SaveFile(bat,
  ":Repeat\r\n"
  "del \"" + path + "\"\r\n"
  "if exist \"" + path + "\" goto Repeat\r\n"
  "rmdir \"" + dir + "\"\r\n"
  "del \"" + bat + "\"\r\n"
 );

 STARTUPINFO si;
 PROCESS_INFORMATION pi;
 ZeroMemory(&si, sizeof(si));
 si.cb = sizeof(si);
 si.dwFlags = STARTF_USESHOWWINDOW;
 si.wShowWindow = SW_HIDE;
 char h[512];
 strcpy(h, bat);
 CreateProcess(NULL, h, NULL, NULL, FALSE,
               IDLE_PRIORITY_CLASS, NULL, GetShellFolder("Desktop", HKEY_CURRENT_USER),
               &si, &pi);
}

void InstallUninstall(const char *name, const char *dname, const char *cmdline)
{
 String path = "SOFTWARE\\Microsoft\\Windows\\CurrentVersion\\Uninstall\\" + String(name);
 SetWinRegString(dname, "DisplayName", path);
 SetWinRegString(cmdline, "UninstallString", path);
}

bool Install(const char *folder, const char *exe, const char *desc)
{
 String lnk = String(desc) + ".lnk";
 const Vector<String>& arg = CommandLine();
 for(int i = 0; i < arg.GetCount(); i++)
  if(arg[i] == "--uninstall") {
   Uninstall(folder, desc);
   return true;
  }
```

```
String exefile = GetExeFilePath();

char windows_path[MAX_PATH];
GetWindowsDirectory(windows_path, MAX_PATH);

String program_files(windows_path[0], 1);
program_files << ":\\Program Files";
String target = program_files + "\\" + folder + "\\" + exe;

if(strcmpi(exefile, target)) {
 Progress pi;
 pi.SetText("Instaluji program..");
 pi.SetTotal((int)(GetFileLength(exefile) / 16384 - 1));
 pi.Create();
 CreateDirectory(program_files, NULL);
 CreateDirectory(program_files + "\\" + folder, NULL);
 FileIn in(exefile);
 FileOut out(target);
 if(!in || !out) {
  Exclamation("Program nelze nainstalovat.");
  return true;
 }

 int n;
 Buffer<byte> b(16384);
 for(;;) {
  pi.StepCanceled();
  n = in.Get(b, 16384);
  if(n <= 0) break;
  out.Put(b, n);
  if(out.IsError()) break;
 }

 out.Close();

 if(in.IsError() || out.IsError()) {
  Exclamation("Program nelze nainstalovat.");
  return true;
 }

 String desktop_path = GetShellFolder("Desktop", HKEY_CURRENT_USER);

 String linkpath = AppendFileName(desktop_path, lnk);

 HRESULT hres;
 IShellLink* psl;
 IPersistFile* ppf;
```

```
  CoInitialize(NULL);
  hres = CoCreateInstance(CLSID_ShellLink, NULL, CLSCTX_INPROC_SERVER, IID_IShellLink,
                          (PVOID *) &psl);
  const char *pszLinkFilePathname = linkpath;
  if(SUCCEEDED(hres)) {
   psl->SetPath(target);
   psl->SetDescription(desc);
   hres = psl->QueryInterface(IID_IPersistFile, (PVOID *) &ppf);
   if (SUCCEEDED(hres)) {
   #ifndef UNICODE
    WCHAR szPath[_MAX_PATH] = { 0 };
    MultiByteToWideChar(CP_ACP, 0, pszLinkFilePathname,
                        strlen(pszLinkFilePathname), szPath, _MAX_PATH);
    hres = ppf->Save(szPath, TRUE);
   #else
    hres = ppf->Save(pszLinkFilePathname, TRUE);
   #endif
    ppf->Release();
   }
  }
  psl->Release();
  CoUninitialize();

  InstallUninstall(folder, desc, target + " --uninstall");




  WinExec(target, SW_SHOWNORMAL);
  return true;
 }
 return false;
}
```

In GUI_APP_MAIN, you use it like this:

```
GUI_APP_MAIN
{
 if(Install("my_app_folder_and_id", "MyApp.exe", "My application name")) return;
}
```