sdk32 Module Manual

Release 0.03

Object Craft

April 24, 2003

E-mail: djc@object-craft.com.au

Abstract

The sdk32 module is a very thin partial wrapper of the Win32 Platform SDK. It is intended to be used in embedding Python in non-MFC Win32 applications.

See Also:

sdk32 Module Web Site
(http://www.object-craft.com.au/projects/sdk32/)
for information on sdk32 module

Contents

`string_to_ptr(`*str*`)`

Wraps the `string_to_ptr()` function. The C signature of `string_to_ptr()` is

```
PyObject string_to_ptr(char *str)
```

The function is called like this:

```
ret = PyInt_FromLong((int)str);
```

Returns ret.

`string_from_ptr(`*ptr*`)`

Wraps the `string_from_ptr()` function. The C signature of `string_from_ptr()` is

```
PyObject string_from_ptr(int ptr)
```

The function is called like this:

```
ret = PyString_FromString((char *)ptr);
```

Returns ret.

`Buffer()`

Wraps the `Buffer()` function. The C signature of `Buffer()` is

```
Buffer Buffer()
```

The function is called like this:

```
ret = Buffer();
```

Returns ret.

`open_debug_log(`*fname, mode*`)`

Wraps the `open_debug_log()` function. The C signature of `open_debug_log()` is

```
void open_debug_log(char *fname, char *mode)
```

The function is called like this:

```
open_debug_log(fname, mode);
```

`MAKE_DIALOG_TEMPLATE(`*seq*`)`

Wraps the `MAKE_DIALOG_TEMPLATE()` function. The C signature of `MAKE_DIALOG_TEMPLATE()` is

```
DlgTempl MAKE_DIALOG_TEMPLATE(PyObject *seq)
```

The function is called like this:

```
ret = MAKE_DIALOG_TEMPLATE(seq);
```

Returns ret.

`GlobalMemoryStatus`(*buff*)
Wraps the `GlobalMemoryStatus()` function. The C signature of `GlobalMemoryStatus()` is

```
VOID GlobalMemoryStatus(MEMORYSTATUS *buff)
```

The function is called like this:

```
GlobalMemoryStatus(&buff->mem);
```

`GetProfileInt`(*app_name, key_name, defval*)
Wraps the `GetProfileInt()` function. The C signature of `GetProfileInt()` is

```
UINT GetProfileInt(LPCSTR app_name, LPCSTR key_name, INT defval)
```

The function is called like this:

```
ret = GetProfileInt((LPCSTR)app_name, (LPCSTR)key_name, (INT)defval);
```

Returns ret.

`GetProfileString`(*app_name, key_name, defval, val, val_len*)
Wraps the `GetProfileString()` function. The C signature of `GetProfileString()` is

```
DWORD GetProfileString(LPCSTR app_name, LPCSTR key_name, LPCSTR defval, LPSTR val, int val_len)
```

The function is called like this:

```
ret = GetProfileString((LPCSTR)app_name, (LPCSTR)key_name, (LPCSTR)defval, val->buff, val_len);
```

Returns ret, str.

`GetTickCount`()
Wraps the `GetTickCount()` function. The C signature of `GetTickCount()` is

```
DWORD GetTickCount()
```

The function is called like this:

```
ret = GetTickCount();
```

Returns ret.

`Sleep`(*millisecs*)
Wraps the `Sleep()` function. The C signature of `Sleep()` is

```
VOID Sleep(DWORD millisecs)
```

The function is called like this:

```
Sleep((DWORD)millisecs);
```

`AppendMenu`(*menu, flags, new_item_id, new_item*)
Wraps the `AppendMenu()` function. The C signature of `AppendMenu()` is

```
BOOL AppendMenu(HMENU menu, UINT flags, UINT new_item_id, LPCSTR new_item)
```

The function is called like this:

```
ret = AppendMenu((HMENU)menu, (UINT)flags, (UINT)new_item_id, (LPCSTR)new_item);
```

Returns ret.

`CallWindowProc(`*wnd_proc, wnd, msg, wparam, lparam*`)`
Wraps the `CallWindowProc()` function. The C signature of `CallWindowProc()` is

```
LRESULT CallWindowProc(WNDPROC wnd_proc, HWND wnd, UINT msg, WPARAM wparam, LPARAM lparam)
```

The function is called like this:

```
ret = wnd_proc_call(wnd_proc, wnd, msg, wparam, lparam);
```

Returns ret.

`ClientToScreen(`*wnd, point*`)`
Wraps the `ClientToScreen()` function. The C signature of `ClientToScreen()` is

```
BOOL ClientToScreen(HWND wnd, POINT *point)
```

The function is called like this:

```
ret = ClientToScreen((HWND)wnd, &point->point);
```

Returns ret.

`CheckMenuItem(`*menu, item_id, check*`)`
Wraps the `CheckMenuItem()` function. The C signature of `CheckMenuItem()` is

```
DWORD CheckMenuItem(HMENU menu, UINT item_id, UINT check)
```

The function is called like this:

```
ret = CheckMenuItem((HMENU)menu, (UINT)item_id, (UINT)check);
```

Returns ret.

`CheckMenuRadioItem(`*menu, first_id, last_id, check_id, flags*`)`
Wraps the `CheckMenuRadioItem()` function. The C signature of `CheckMenuRadioItem()` is

```
BOOL CheckMenuRadioItem(HMENU menu, UINT first_id, UINT last_id, UINT check_id, UINT flags)
```

The function is called like this:

```
ret = CheckMenuRadioItem((HMENU)menu, (UINT)first_id, (UINT)last_id, (UINT)check_id, (UINT)flags);
```

Returns ret.

`CreateMenu()`
Wraps the `CreateMenu()` function. The C signature of `CreateMenu()` is

Index

```
ret = WSAGETSELECTEVENT(lparam);
```

Returns ret.

`WSAGETSELECTERROR(`*lparam*`)`
Wraps the `WSAGETSELECTERROR()` function. The C signature of `WSAGETSELECTERROR()` is

```
int WSAGETSELECTERROR(int lparam)
```

The function is called like this:

```
ret = WSAGETSELECTERROR(lparam);
```

Returns ret.


```
HMENU CreateMenu()
```

The function is called like this:

```
ret = CreateMenu();
```

Returns ret.

`CreatePopupMenu()`
Wraps the `CreatePopupMenu()` function. The C signature of `CreatePopupMenu()` is

```
HMENU CreatePopupMenu()
```

The function is called like this:

```
ret = CreatePopupMenu();
```

Returns ret.

`CreateWindow(`*class_name, window_name, style, x, y, width, height, parent, menu, inst, param*`)`
Wraps the `CreateWindow()` function. The C signature of `CreateWindow()` is

```
HWND CreateWindow(LPCSTR class_name, LPCSTR window_name, DWORD style, int x, int y, int width, int h
```

The function is called like this:

```
ret = CreateWindow((LPCSTR)class_name, window_name, (DWORD)style, x, y, width, height, (HWND)parent,
```

Returns ret.

`DefWindowProc(`*wnd, msg, wparam, lparam*`)`
Wraps the `DefWindowProc()` function. The C signature of `DefWindowProc()` is

```
LRESULT DefWindowProc(HWND wnd, UINT msg, WPARAM wparam, LPARAM lparam)
```

The function is called like this:

```
ret = DefWindowProc((HWND)wnd, (UINT)msg, (WPARAM)wparam, (LPARAM)lparam);
```

Returns ret.

`DestroyCaret()`
Wraps the `DestroyCaret()` function. The C signature of `DestroyCaret()` is

```
BOOL DestroyCaret()
```

The function is called like this:

```
ret = DestroyCaret();
```

Returns ret.

`DestroyMenu(`*menu*`)`
Wraps the `DestroyMenu()` function. The C signature of `DestroyMenu()` is

```
BOOL DestroyMenu(HMENU menu)
```

The function is called like this:

```
ret = DestroyMenu((HMENU)menu);
```

Returns ret.

`DestroyWindow(`*wnd*`)`
Wraps the `DestroyWindow()` function. The C signature of `DestroyWindow()` is

```
BOOL DestroyWindow(HWND wnd)
```

The function is called like this:

```
ret = DestroyWindow((HWND)wnd);
```

Returns ret.

`DialogBox(`*inst, templ_name, parent_wnd, dlg_proc*`)`
Wraps the `DialogBox()` function. The C signature of `DialogBox()` is

```
int DialogBox(HINSTANCE inst, LPCSTR templ_name, HWND parent_wnd, DLGPROC dlg_proc)
```

The function is called like this:

```
ret = DialogBoxParam(
    (HINSTANCE)inst, (LPCSTR)templ_name,
    (HWND)parent_wnd, (DLGPROC)dlg_proc_trampoline,
    (LPARAM)dlg_proc);
```

Returns ret.

`DialogBoxIndirect(`*inst, templ, parent_wnd, dlg_proc*`)`
Wraps the `DialogBoxIndirect()` function. The C signature of `DialogBoxIndirect()` is

```
int DialogBoxIndirect(HINSTANCE inst, void *templ, HWND parent_wnd, DLGPROC dlg_proc)
```

The function is called like this:

```
ret = DialogBoxIndirectParam(
    (HINSTANCE)inst, (LPCDLGTEMPLATE)templ,
    (HWND)parent_wnd, (DLGPROC)dlg_proc_trampoline,
    (LPARAM)dlg_proc);
```

Returns ret.

`DispatchMessage(`*msg*`)`
Wraps the `DispatchMessage()` function. The C signature of `DispatchMessage()` is

```
LRESULT DispatchMessage(MSG *msg)
```

The function is called like this:


```
ret = WSACancelAsyncRequest((HANDLE)async_handle);
```

Returns ret.

`WSACleanup()`
Wraps the `WSACleanup()` function. The C signature of `WSACleanup()` is

```
int WSACleanup()
```

The function is called like this:

```
ret = WSACleanup();
```

Returns ret.

`WSAGetLastError()`
Wraps the `WSAGetLastError()` function. The C signature of `WSAGetLastError()` is

```
int WSAGetLastError()
```

The function is called like this:

```
ret = WSAGetLastError();
```

Returns ret.

`WSAStartup(`*version, wsa_data*`)`
Wraps the `WSAStartup()` function. The C signature of `WSAStartup()` is

```
int WSAStartup(WORD version, WSADATA *wsa_data)
```

The function is called like this:

```
ret = WSAStartup((WORD)version, &wsa_data->data);
```

Returns ret.

`WSAGETASYNCERROR(`*lparam*`)`
Wraps the `WSAGETASYNCERROR()` function. The C signature of `WSAGETASYNCERROR()` is

```
int WSAGETASYNCERROR(int lparam)
```

The function is called like this:

```
ret = WSAGETASYNCERROR(lparam);
```

Returns ret.

`WSAGETSELECTEVENT(`*lparam*`)`
Wraps the `WSAGETSELECTEVENT()` function. The C signature of `WSAGETSELECTEVENT()` is

```
int WSAGETSELECTEVENT(int lparam)
```

The function is called like this:

```
ret = socket(af, type, protocol);
```

Returns ret.

`WSAAsyncGetHostByAddr`(*wnd, msg, addr, len, type, buf, buflen*)
Wraps the `WSAAsyncGetHostByAddr()` function. The C signature of `WSAAsyncGetHostByAddr()` is

```
HANDLE WSAAsyncGetHostByAddr(HWND wnd, u_int msg, char *addr, int len, int type, char *buf, int bufl
```

The function is called like this:

```
ret = WSAAsyncGetHostByAddr((HWND)wnd, (u_int)msg, addr, len, type, buf->buff, buflen);
```

Returns ret, str.

`WSAAsyncGetHostByName`(*wnd, msg, name, buf, buflen*)
Wraps the `WSAAsyncGetHostByName()` function. The C signature of `WSAAsyncGetHostByName()` is

```
HANDLE WSAAsyncGetHostByName(HWND wnd, u_int msg, char *name, char *buf, int buflen)
```

The function is called like this:

```
ret = WSAAsyncGetHostByName((HWND)wnd, (u_int)msg, name, buf->buff, buflen);
```

Returns ret, str.

`WSAAsyncGetServByName`(*wnd, msg, name, proto, buf, buflen*)
Wraps the `WSAAsyncGetServByName()` function. The C signature of `WSAAsyncGetServByName()` is

```
HANDLE WSAAsyncGetServByName(HWND wnd, u_int msg, char *name, char *proto, char *buf, int buflen)
```

The function is called like this:

```
ret = WSAAsyncGetServByName((HWND)wnd, (u_int)msg, name, proto, buf->buff, buflen);
```

Returns ret, str.

`WSAAsyncSelect`(*s, wnd, msg, event*)
Wraps the `WSAAsyncSelect()` function. The C signature of `WSAAsyncSelect()` is

```
int WSAAsyncSelect(SOCKET s, HWND wnd, UINT msg, long event)
```

The function is called like this:

```
ret = WSAAsyncSelect((SOCKET)s, (HWND)wnd, (UINT)msg, (long)event);
```

Returns ret.

`WSACancelAsyncRequest`(*async_handle*)
Wraps the `WSACancelAsyncRequest()` function. The C signature of `WSACancelAsyncRequest()` is

```
int WSACancelAsyncRequest(HANDLE async_handle)
```

The function is called like this:


```
ret = DispatchMessage(&msg->msg);
```

Returns ret.

`EnableMenuItem`(*menu, enable_item_id, enable*)
Wraps the `EnableMenuItem()` function. The C signature of `EnableMenuItem()` is

```
BOOL EnableMenuItem(HMENU menu, UINT enable_item_id, UINT enable)
```

The function is called like this:

```
ret = EnableMenuItem((HMENU)menu, (UINT)enable_item_id, (UINT)enable);
```

Returns ret.

`EndDialog`(*dlg, result*)
Wraps the `EndDialog()` function. The C signature of `EndDialog()` is

```
BOOL EndDialog(HWND dlg, int result)
```

The function is called like this:

```
ret = EndDialog((HWND)dlg, result);
```

Returns ret.

`GetClassLong`(*wnd, index*)
Wraps the `GetClassLong()` function. The C signature of `GetClassLong()` is

```
ULONG GetClassLong(HWND wnd, int index)
```

The function is called like this:

```
ret = name_proc_getclasslong(wnd, index);
```

Returns ret.

`GetClientRect`(*wnd, rect*)
Wraps the `GetClientRect()` function. The C signature of `GetClientRect()` is

```
BOOL GetClientRect(HWND wnd, RECT *rect)
```

The function is called like this:

```
ret = GetClientRect((HWND)wnd, &rect->rect);
```

Returns ret.

`GetCursorPos`(*point*)
Wraps the `GetCursorPos()` function. The C signature of `GetCursorPos()` is

```
BOOL GetCursorPos(POINT *point)
```

The function is called like this:

```
ret = GetCursorPos(&point->point);
```

Returns ret.

`GetDlgItemInt`(*dlg, id, trans, is_signed*)
Wraps the `GetDlgItemInt()` function. The C signature of `GetDlgItemInt()` is

```
UINT GetDlgItemInt(HWND dlg, int id, BOOL *trans, BOOL is_signed)
```

The function is called like this:

```
ret = GetDlgItemInt((HWND)dlg, id, &trans, (BOOL)is_signed);
```

Returns ret, trans.

`GetDlgItemText`(*dlg, id, val, max_val*)
Wraps the `GetDlgItemText()` function. The C signature of `GetDlgItemText()` is

```
UINT GetDlgItemText(HWND dlg, int id, LPSTR val, int max_val)
```

The function is called like this:

```
ret = GetDlgItemText((HWND)dlg, id, val->buff, max_val);
```

Returns ret, str.

`GetFocus()`
Wraps the `GetFocus()` function. The C signature of `GetFocus()` is

```
HWND GetFocus()
```

The function is called like this:

```
ret = GetFocus();
```

Returns ret.

`GetKeyState`(*virt_key*)
Wraps the `GetKeyState()` function. The C signature of `GetKeyState()` is

```
SHORT GetKeyState(int virt_key)
```

The function is called like this:

```
ret = GetKeyState(virt_key);
```

Returns ret.

`GetMessage`(*msg, wnd, filter_min, filter_max*)
Wraps the `GetMessage()` function. The C signature of `GetMessage()` is

```
BOOL GetMessage(MSG *msg, HWND wnd, UINT filter_min, UINT filter_max)
```

The function is called like this:


```
ret = inet_addr(str);
```

Returns ret.

`inet_ntoa`(*addr*)
Wraps the `inet_ntoa()` function. The C signature of `inet_ntoa()` is

```
char *inet_ntoa(int addr)
```

The function is called like this:

```
ret = inet_ntoa(addr);
```

Returns ret.

`ioctlsocket`(*s, cmd, argp*)
Wraps the `ioctlsocket()` function. The C signature of `ioctlsocket()` is

```
int ioctlsocket(SOCKET s, long cmd, u_long *argp)
```

The function is called like this:

```
ret = ioctlsocket((SOCKET)s, (long)cmd, &argp);
```

Returns ret, argp.

`recv`(*s, buf, len, flags*)
Wraps the `recv()` function. The C signature of `recv()` is

```
int recv(SOCKET s, LPSTR buf, int len, int flags)
```

The function is called like this:

```
ret = recv((SOCKET)s, buf->buff, len, flags);
```

Returns ret, str.

`send`(*s, buf, len, flags*)
Wraps the `send()` function. The C signature of `send()` is

```
int send(SOCKET s, LPSTR buf, int len, int flags)
```

The function is called like this:

```
ret = send((SOCKET)s, (LPSTR)buf, len, flags);
```

Returns ret.

`socket`(*af, type, protocol*)
Wraps the `socket()` function. The C signature of `socket()` is

```
SOCKET socket(int af, int type, int protocol)
```

The function is called like this:

```
ret = connect((SOCKET)s, (SOCKADDR*)&addr->addr, sizeof(addr->addr));
```

Returns ret.

`ntohl`(*val*)
Wraps the `ntohl()` function. The C signature of `ntohl()` is

```
u_long ntohl(u_long val)
```

The function is called like this:

```
ret = ntohl((u_long)val);
```

Returns ret.

`htonl`(*val*)
Wraps the `htonl()` function. The C signature of `htonl()` is

```
u_long htonl(u_long val)
```

The function is called like this:

```
ret = htonl((u_long)val);
```

Returns ret.

`htons`(*val*)
Wraps the `htons()` function. The C signature of `htons()` is

```
u_short htons(u_short val)
```

The function is called like this:

```
ret = htons((u_short)val);
```

Returns ret.

`ntohs`(*val*)
Wraps the `ntohs()` function. The C signature of `ntohs()` is

```
u_short ntohs(u_short val)
```

The function is called like this:

```
ret = ntohs((u_short)val);
```

Returns ret.

`inet_addr`(*str*)
Wraps the `inet_addr()` function. The C signature of `inet_addr()` is

```
u_long inet_addr(char *str)
```

The function is called like this:


```
ret = GetMessage(&msg->msg, (HWND)wnd, (UINT)filter_min, (UINT)filter_max);
```

Returns ret.

`GetModuleHandle`(*name*)
Wraps the `GetModuleHandle()` function. The C signature of `GetModuleHandle()` is

```
HMODULE GetModuleHandle(LPCSTR name)
```

The function is called like this:

```
ret = GetModuleHandle(name);
```

Returns ret.

`GetParent`(*wnd*)
Wraps the `GetParent()` function. The C signature of `GetParent()` is

```
HWND GetParent(HWND wnd)
```

The function is called like this:

```
ret = GetParent((HWND)wnd);
```

Returns ret.

`GetPrivateProfileInt`(*app, key, defval, file*)
Wraps the `GetPrivateProfileInt()` function. The C signature of `GetPrivateProfileInt()` is

```
UINT GetPrivateProfileInt(LPCSTR app, LPCSTR key, int defval, LPCSTR file)
```

The function is called like this:

```
ret = GetPrivateProfileInt((LPCSTR)app, (LPCSTR)key, defval, (LPCSTR)file);
```

Returns ret.

`GetPrivateProfileString`(*app, key, defval, val, val_len, file*)
Wraps the `GetPrivateProfileString()` function. The C signature of `GetPrivateProfileString()` is

```
DWORD GetPrivateProfileString(LPCSTR app, LPCSTR key, LPCSTR defval, LPSTR val, int val_len, LPCSTR
```

The function is called like this:

```
ret = GetPrivateProfileString((LPCSTR)app, (LPCSTR)key, (LPCSTR)defval, val->buff, val_len, (LPCSTR)
```

Returns ret, str.

`GetScrollPos`(*wnd, bar*)
Wraps the `GetScrollPos()` function. The C signature of `GetScrollPos()` is

```
int GetScrollPos(HWND wnd, int bar)
```

The function is called like this:

```
ret = GetScrollPos((HWND)wnd, bar);
```

Returns ret.

`GetWindowLong(`*wnd, index*`)`
Wraps the `GetWindowLong()` function. The C signature of `GetWindowLong()` is

```
LONGOBJ GetWindowLong(HWND wnd, int index)
```

The function is called like this:

```
ret = wnd_proc_getwindowlong(wnd, index);
```

Returns ret.

`GetSysColor(`*index*`)`
Wraps the `GetSysColor()` function. The C signature of `GetSysColor()` is

```
DWORD GetSysColor(int index)
```

The function is called like this:

```
ret = GetSysColor(index);
```

Returns ret.

`GetSystemMetrics(`*index*`)`
Wraps the `GetSystemMetrics()` function. The C signature of `GetSystemMetrics()` is

```
int GetSystemMetrics(int index)
```

The function is called like this:

```
ret = GetSystemMetrics(index);
```

Returns ret.

`GetWindowRect(`*wnd, rect*`)`
Wraps the `GetWindowRect()` function. The C signature of `GetWindowRect()` is

```
BOOL GetWindowRect(HWND wnd, RECT *rect)
```

The function is called like this:

```
ret = GetWindowRect((HWND)wnd, &rect->rect);
```

Returns ret.

`HideCaret(`*wnd*`)`
Wraps the `HideCaret()` function. The C signature of `HideCaret()` is

```
BOOL HideCaret(HWND wnd)
```

The function is called like this:


```
ret = servent_from_buffer((Buffer_Obj *)buff);
```

Returns ret.

`hostent_from_buffer(`*buff*`)`
Wraps the `hostent_from_buffer()` function. The C signature of `hostent_from_buffer()` is

```
HOSTENT hostent_from_buffer(Buffer *buff)
```

The function is called like this:

```
ret = hostent_from_buffer((Buffer_Obj *)buff);
```

Returns ret.

`buffer_ensure_size(`*buff, size*`)`
Wraps the `buffer_ensure_size()` function. The C signature of `buffer_ensure_size()` is

```
int buffer_ensure_size(Buffer *buff, int size)
```

The function is called like this:

```
ret = buffer_ensure_size((Buffer_Obj *)buff, size);
```

Returns ret.

`buffer_get_ptr(`*buff*`)`
Wraps the `buffer_get_ptr()` function. The C signature of `buffer_get_ptr()` is

```
int buffer_get_ptr(Buffer *buff)
```

The function is called like this:

```
ret = (int)((Buffer_Obj*)buff)->buff;
```

Returns ret.

`closesocket(`*s*`)`
Wraps the `closesocket()` function. The C signature of `closesocket()` is

```
int closesocket(SOCKET s)
```

The function is called like this:

```
ret = closesocket((SOCKET)s);
```

Returns ret.

`connect(`*s, addr*`)`
Wraps the `connect()` function. The C signature of `connect()` is

```
int connect(SOCKET s, SOCKADDR_IN *addr, int namelen)
```

The function is called like this:

```
ret = SetPixel((HDC)dc, x, y, (COLORREF)color);
```

Returns ret.

`SetTextAlign`(*dc, mode*)
Wraps the `SetTextAlign()` function. The C signature of `SetTextAlign()` is

```
UINT SetTextAlign(HDC dc, UINT mode)
```

The function is called like this:

```
ret = SetTextAlign((HDC)dc, (UINT)mode);
```

Returns ret.

`SetTextColor`(*dc, color*)
Wraps the `SetTextColor()` function. The C signature of `SetTextColor()` is

```
COLORREF SetTextColor(HDC dc, COLORREF color)
```

The function is called like this:

```
ret = SetTextColor((HDC)dc, (COLORREF)color);
```

Returns ret.

`TextOut`(*dc, x, y, text, len*)
Wraps the `TextOut()` function. The C signature of `TextOut()` is

```
BOOL TextOut(HDC dc, int x, int y, LPCSTR text, int len)
```

The function is called like this:

```
ret = TextOut((HDC)dc, x, y, (LPCSTR)text, len);
```

Returns ret.

`UnrealizeObject`(*obj*)
Wraps the `UnrealizeObject()` function. The C signature of `UnrealizeObject()` is

```
BOOL UnrealizeObject(HGDIOBJ obj)
```

The function is called like this:

```
ret = UnrealizeObject((HGDIOBJ)obj);
```

Returns ret.

`servent_from_buffer`(*buff*)
Wraps the `servent_from_buffer()` function. The C signature of `servent_from_buffer()` is

```
SERVENT servent_from_buffer(Buffer *buff)
```

The function is called like this:


```
ret = HideCaret((HWND)wnd);
```

Returns ret.

`HIWORD`(*value*)
Wraps the `HIWORD()` function. The C signature of `HIWORD()` is

```
int HIWORD(int value)
```

The function is called like this:

```
ret = HIWORD(value);
```

Returns ret.

`IsDialogMessage`(*dlg, msg*)
Wraps the `IsDialogMessage()` function. The C signature of `IsDialogMessage()` is

```
BOOL IsDialogMessage(HWND dlg, MSG *msg)
```

The function is called like this:

```
ret = IsDialogMessage((HWND)dlg, &msg->msg);
```

Returns ret.

`KillTimer`(*wnd, event*)
Wraps the `KillTimer()` function. The C signature of `KillTimer()` is

```
BOOL KillTimer(HWND wnd, UINT event)
```

The function is called like this:

```
timerproc_remove((HWND)wnd, (UINT)event);
ret = KillTimer((HWND)wnd, (UINT)event);
```

Returns ret.

`LoadAccelerators`(*inst, table_name*)
Wraps the `LoadAccelerators()` function. The C signature of `LoadAccelerators()` is

```
HACCEL LoadAccelerators(HINSTANCE inst, LPCSTR table_name)
```

The function is called like this:

```
ret = LoadAccelerators((HINSTANCE)inst, table_name);
```

Returns ret.

`LoadCursor`(*inst, name*)
Wraps the `LoadCursor()` function. The C signature of `LoadCursor()` is

```
HCURSOR LoadCursor(HINSTANCE inst, LPCSTR name)
```

The function is called like this:

```
ret = LoadCursor((HINSTANCE)inst, name);
```

Returns ret.

`LoadIcon`(*inst, name*)
Wraps the `LoadIcon()` function. The C signature of `LoadIcon()` is

```
HICON LoadIcon(HINSTANCE inst, LPCSTR name)
```

The function is called like this:

```
ret = LoadIcon((HINSTANCE)inst, name);
```

Returns ret.

`LoadString`(*inst, id, str, str_len*)
Wraps the `LoadString()` function. The C signature of `LoadString()` is

```
int LoadString(HINSTANCE inst, UINT id, LPSTR str, int str_len)
```

The function is called like this:

```
ret = LoadString((HINSTANCE)inst, (UINT)id, str->buff, str_len);
```

Returns ret, str.

`LOWORD`(*w*)
Wraps the `LOWORD()` function. The C signature of `LOWORD()` is

```
int LOWORD(int w)
```

The function is called like this:

```
ret = LOWORD(w);
```

Returns ret.

`MAKELONG`(*low, high*)
Wraps the `MAKELONG()` function. The C signature of `MAKELONG()` is

```
DWORD MAKELONG(WORD low, WORD high)
```

The function is called like this:

```
ret = MAKELONG((WORD)low, (WORD)high);
```

Returns ret.

`MessageBeep`(*type*)
Wraps the `MessageBeep()` function. The C signature of `MessageBeep()` is

```
BOOL MessageBeep(UINT type)
```

The function is called like this:


```
ret = RGB((BYTE)red, (BYTE)green, (BYTE)blue);
```

Returns ret.

`SelectObject`(*dc, obj*)
Wraps the `SelectObject()` function. The C signature of `SelectObject()` is

```
HGDIOBJ SelectObject(HDC dc, HGDIOBJ obj)
```

The function is called like this:

```
ret = SelectObject((HDC)dc, (HGDIOBJ)obj);
```

Returns ret.

`SetBkColor`(*dc, color*)
Wraps the `SetBkColor()` function. The C signature of `SetBkColor()` is

```
COLORREF SetBkColor(HDC dc, COLORREF color)
```

The function is called like this:

```
ret = SetBkColor((HDC)dc, (COLORREF)color);
```

Returns ret.

`SetBkMode`(*dc, mode*)
Wraps the `SetBkMode()` function. The C signature of `SetBkMode()` is

```
int SetBkMode(HDC dc, int mode)
```

The function is called like this:

```
ret = SetBkMode((HDC)dc, mode);
```

Returns ret.

`SetBrushOrgEx`(*dc, x_org, y_org, pt*)
Wraps the `SetBrushOrgEx()` function. The C signature of `SetBrushOrgEx()` is

```
BOOL SetBrushOrgEx(HDC dc, int x_org, int y_org, POINT *pt)
```

The function is called like this:

```
ret = SetBrushOrgEx((HDC)dc, x_org, y_org, pt);
```

Returns ret.

`SetPixel`(*dc, x, y, color*)
Wraps the `SetPixel()` function. The C signature of `SetPixel()` is

```
COLORREF SetPixel(HDC dc, int x, int y, COLORREF color)
```

The function is called like this:

```
ret = InvalidateRect((HWND)wnd, rect, (BOOL)erase);
```

Returns ret.

`LineTo`(*dc, x, y*)

Wraps the `LineTo()` function. The C signature of `LineTo()` is

```
BOOL LineTo(HDC dc, int x, int y)
```

The function is called like this:

```
ret = LineTo((HDC)dc, x, y);
```

Returns ret.

`MoveToEx`(*dc, x, y, point*)

Wraps the `MoveToEx()` function. The C signature of `MoveToEx()` is

```
BOOL MoveToEx(HDC dc, int x, int y, POINT *point)
```

The function is called like this:

```
ret = MoveToEx((HDC)dc, x, y, point);
```

Returns ret.

`Rectangle`(*dc, left, top, right, bottom*)

Wraps the `Rectangle()` function. The C signature of `Rectangle()` is

```
BOOL Rectangle(HDC dc, int left, int top, int right, int bottom)
```

The function is called like this:

```
ret = Rectangle((HDC)dc, left, top, right, bottom);
```

Returns ret.

`ReleaseDC`(*wnd, dc*)

Wraps the `ReleaseDC()` function. The C signature of `ReleaseDC()` is

```
int ReleaseDC(HWND wnd, HDC dc)
```

The function is called like this:

```
ret = ReleaseDC((HWND)wnd, (HDC)dc);
```

Returns ret.

`RGB`(*red, green, blue*)

Wraps the `RGB()` function. The C signature of `RGB()` is

```
COLORREF RGB(BYTE red, BYTE green, BYTE blue)
```

The function is called like this:


```
ret = MessageBeep((UINT)type);
```

Returns ret.

`MessageBox`(*wnd, text, title, type*)

Wraps the `MessageBox()` function. The C signature of `MessageBox()` is

```
int MessageBox(HWND wnd, LPCSTR text, LPCSTR title, UINT type)
```

The function is called like this:

```
ret = MessageBox((HWND)wnd, (LPCSTR)text, (LPCSTR)title, (UINT)type);
```

Returns ret.

`MoveWindow`(*wnd, x, y, width, height, repaint*)

Wraps the `MoveWindow()` function. The C signature of `MoveWindow()` is

```
BOOL MoveWindow(HWND wnd, int x, int y, int width, int height, BOOL repaint)
```

The function is called like this:

```
ret = MoveWindow((HWND)wnd, x, y, width, height, (BOOL)repaint);
```

Returns ret.

`PeekMessage`(*msg, wnd, filter_min, filter_max, flags*)

Wraps the `PeekMessage()` function. The C signature of `PeekMessage()` is

```
BOOL PeekMessage(MSG *msg, HWND wnd, UINT filter_min, UINT filter_max, UINT flags)
```

The function is called like this:

```
ret = PeekMessage(&msg->msg, (HWND)wnd, (UINT)filter_min, (UINT)filter_max, (UINT)flags);
```

Returns ret.

`PostMessage`(*wnd, msg, wparam, lparam*)

Wraps the `PostMessage()` function. The C signature of `PostMessage()` is

```
BOOL PostMessage(HWND wnd, UINT msg, WPARAM wparam, LPARAM lparam)
```

The function is called like this:

```
ret = PostMessage((HWND)wnd, (UINT)msg, (WPARAM)wparam, (LPARAM)lparam);
```

Returns ret.

`PostQuitMessage`(*exit_code*)

Wraps the `PostQuitMessage()` function. The C signature of `PostQuitMessage()` is

```
void PostQuitMessage(int exit_code)
```

The function is called like this:

```
PostQuitMessage(exit_code);
```

`RegisterClassEx(`*wc*`)`

Wraps the `RegisterClassEx()` function. The C signature of `RegisterClassEx()` is

```
ATOM RegisterClassEx(WNDCLASSEX *wc)
```

The function is called like this:

```
if (!name_proc_add(wc->wc.lpszClassName, wc->proxy_wnd_proc))
    return NULL;
ret = RegisterClassEx(&wc->wc);
```

Returns ret.

`ScreenToClient(`*wnd, point*`)`

Wraps the `ScreenToClient()` function. The C signature of `ScreenToClient()` is

```
BOOL ScreenToClient(HWND wnd, POINT *point)
```

The function is called like this:

```
ret = ScreenToClient((HWND)wnd, &point->point);
```

Returns ret.

`ScrollWindow(`*wnd, x, y, rect, clip*`)`

Wraps the `ScrollWindow()` function. The C signature of `ScrollWindow()` is

```
BOOL ScrollWindow(HWND wnd, int x, int y, RECT *rect, RECT *clip)
```

The function is called like this:

```
ret = ScrollWindow((HWND)wnd, x, y, rect, clip);
```

Returns ret.

`SendDlgItemMessage(`*dlg, id, msg, wparam, lparam*`)`

Wraps the `SendDlgItemMessage()` function. The C signature of `SendDlgItemMessage()` is

```
LRESULT SendDlgItemMessage(HWND dlg, int id, UINT msg, WPARAM wparam, LPARAM lparam)
```

The function is called like this:

```
ret = SendDlgItemMessage((HWND)dlg, id, (UINT)msg, (WPARAM)wparam, (LPARAM)lparam);
```

Returns ret.

`SendMessage(`*wnd, msg, wparam, lparam*`)`

Wraps the `SendMessage()` function. The C signature of `SendMessage()` is

```
LRESULT SendMessage(HWND wnd, UINT msg, WPARAM wparam, LPARAM lparam)
```

The function is called like this:


```
ret = GetDC((HWND)wnd);
```

Returns ret.

`GetDeviceCaps(`*dc, index*`)`

Wraps the `GetDeviceCaps()` function. The C signature of `GetDeviceCaps()` is

```
int GetDeviceCaps(HDC dc, int index)
```

The function is called like this:

```
ret = GetDeviceCaps((HDC)dc, index);
```

Returns ret.

`CreateIC(`*driver, device, output*`)`

Wraps the `CreateIC()` function. The C signature of `CreateIC()` is

```
HDC CreateIC(LPCSTR driver, LPCSTR device, LPCSTR output, DEVMODE *init)
```

The function is called like this:

```
ret = CreateIC((LPCSTR)driver, device, output, NULL);
```

Returns ret.

`GetStockObject(`*object*`)`

Wraps the `GetStockObject()` function. The C signature of `GetStockObject()` is

```
HGDIOBJ GetStockObject(int object)
```

The function is called like this:

```
ret = GetStockObject(object);
```

Returns ret.

`GetTextMetrics(`*dc, tm*`)`

Wraps the `GetTextMetrics()` function. The C signature of `GetTextMetrics()` is

```
BOOL GetTextMetrics(HDC dc, TEXTMETRIC *tm)
```

The function is called like this:

```
ret = GetTextMetrics((HDC)dc, &tm->tm);
```

Returns ret.

`InvalidateRect(`*wnd, rect, erase*`)`

Wraps the `InvalidateRect()` function. The C signature of `InvalidateRect()` is

```
BOOL InvalidateRect(HWND wnd, RECT *rect, BOOL erase)
```

The function is called like this:

```
ret = DeleteObject((HGDIOBJ)obj);
```

Returns ret.

`DrawText`(*dc, text, count, rect, fmt*)
Wraps the `DrawText()` function. The C signature of `DrawText()` is

```
int DrawText(HDC dc, LPCSTR text, int count, RECT *rect, UINT fmt)
```

The function is called like this:

```
ret = DrawText((HDC)dc, (LPCSTR)text, count, &rect->rect, (UINT)fmt);
```

Returns ret.

`Ellipse`(*dc, left, top, right, bottom*)
Wraps the `Ellipse()` function. The C signature of `Ellipse()` is

```
BOOL Ellipse(HDC dc, int left, int top, int right, int bottom)
```

The function is called like this:

```
ret = Ellipse((HDC)dc, left, top, right, bottom);
```

Returns ret.

`EndPaint`(*wnd, ps*)
Wraps the `EndPaint()` function. The C signature of `EndPaint()` is

```
BOOL EndPaint(HWND wnd, PAINTSTRUCT *ps)
```

The function is called like this:

```
ret = EndPaint((HWND)wnd, &ps->ps);
```

Returns ret.

`FillRect`(*dc, rect, brush*)
Wraps the `FillRect()` function. The C signature of `FillRect()` is

```
int FillRect(HDC dc, RECT *rect, HBRUSH brush)
```

The function is called like this:

```
ret = FillRect((HDC)dc, &rect->rect, (HBRUSH)brush);
```

Returns ret.

`GetDC`(*wnd*)
Wraps the `GetDC()` function. The C signature of `GetDC()` is

```
HDC GetDC(HWND wnd)
```

The function is called like this:


```
ret = SendMessage((HWND)wnd, (UINT)msg, (WPARAM)wparam, (LPARAM)lparam);
```

Returns ret.

`SetCaretPos`(*x, y*)
Wraps the `SetCaretPos()` function. The C signature of `SetCaretPos()` is

```
BOOL SetCaretPos(int x, int y)
```

The function is called like this:

```
ret = SetCaretPos(x, y);
```

Returns ret.

`SetClassLong`(*wnd, index, new_long*)
Wraps the `SetClassLong()` function. The C signature of `SetClassLong()` is

```
ULONG SetClassLong(HWND wnd, int index, LONG new_long)
```

The function is called like this:

```
ret = name_proc_setclasslong(wnd, index, new_long);
```

Returns ret.

`SetCursorPos`(*x, y*)
Wraps the `SetCursorPos()` function. The C signature of `SetCursorPos()` is

```
BOOL SetCursorPos(int x, int y)
```

The function is called like this:

```
ret = SetCursorPos(x, y);
```

Returns ret.

`SetDlgItemInt`(*dlg, id, val, is_signed*)
Wraps the `SetDlgItemInt()` function. The C signature of `SetDlgItemInt()` is

```
BOOL SetDlgItemInt(HWND dlg, int id, UINT val, BOOL is_signed)
```

The function is called like this:

```
ret = SetDlgItemInt((HWND)dlg, id, (UINT)val, (BOOL)is_signed);
```

Returns ret.

`SetDlgItemText`(*dlg, id, text*)
Wraps the `SetDlgItemText()` function. The C signature of `SetDlgItemText()` is

```
BOOL SetDlgItemText(HWND dlg, int id, LPCSTR text)
```

The function is called like this:

```
ret = SetDlgItemText((HWND)dlg, id, (LPCSTR)text);
```

Returns ret.

`SetFocus`(*wnd*)
Wraps the `SetFocus()` function. The C signature of `SetFocus()` is

```
HWND SetFocus(HWND wnd)
```

The function is called like this:

```
ret = SetFocus((HWND)wnd);
```

Returns ret.

`SetScrollPos`(*wnd, bar, pos, redraw*)
Wraps the `SetScrollPos()` function. The C signature of `SetScrollPos()` is

```
int SetScrollPos(HWND wnd, int bar, int pos, BOOL redraw)
```

The function is called like this:

```
ret = SetScrollPos((HWND)wnd, bar, pos, (BOOL)redraw);
```

Returns ret.

`SetScrollRange`(*wnd, bar, min_pos, max_pos, redraw*)
Wraps the `SetScrollRange()` function. The C signature of `SetScrollRange()` is

```
BOOL SetScrollRange(HWND wnd, int bar, int min_pos, int max_pos, BOOL redraw)
```

The function is called like this:

```
ret = SetScrollRange((HWND)wnd, bar, min_pos, max_pos, (BOOL)redraw);
```

Returns ret.

`SetTimer`(*wnd, timer_id, elapse, timer_func*)
Wraps the `SetTimer()` function. The C signature of `SetTimer()` is

```
UINT SetTimer(HWND wnd, UINT timer_id, UINT elapse, TIMERPROC timer_func)
```

The function is called like this:

```
if (timer_func == Py_None)
    ret = SetTimer((HWND)wnd, (UINT)timer_id, (UINT)elapse, NULL);
else {
    if (timerproc_add((HWND)wnd, (UINT)timer_id, timer_func) < 0)
        return NULL;
    ret = SetTimer((HWND)wnd, (UINT)timer_id, (UINT)elapse, timerproc_trampoline);
}
```

Returns ret.


```
ret = CreateDC((LPCSTR)driver, device, NULL, NULL);
```

Returns ret.

`CreateFont`(*height, width, escapement, orientation, weight, italic, underline, strikeout, charset, output_precision, clip_*
Wraps the `CreateFont()` function. The C signature of `CreateFont()` is

```
HFONT CreateFont(int height, int width, int escapement, int orientation, int weight, DWORD italic, D
```

The function is called like this:

```
ret = CreateFont(height, width, escapement, orientation, weight, (DWORD)italic, (DWORD)underline, (D
```

Returns ret.

`CreateHatchBrush`(*style, color*)
Wraps the `CreateHatchBrush()` function. The C signature of `CreateHatchBrush()` is

```
HBRUSH CreateHatchBrush(int style, COLORREF color)
```

The function is called like this:

```
ret = CreateHatchBrush(style, (COLORREF)color);
```

Returns ret.

`CreateSolidBrush`(*color*)
Wraps the `CreateSolidBrush()` function. The C signature of `CreateSolidBrush()` is

```
HBRUSH CreateSolidBrush(COLORREF color)
```

The function is called like this:

```
ret = CreateSolidBrush((COLORREF)color);
```

Returns ret.

`DeleteDC`(*dc*)
Wraps the `DeleteDC()` function. The C signature of `DeleteDC()` is

```
BOOL DeleteDC(HDC dc)
```

The function is called like this:

```
ret = DeleteDC((HDC)dc);
```

Returns ret.

`DeleteObject`(*obj*)
Wraps the `DeleteObject()` function. The C signature of `DeleteObject()` is

```
BOOL DeleteObject(HGDIOBJ obj)
```

The function is called like this:

```
ret = BeginPaint((HWND)wnd, &ps->ps);
```

Returns ret.

`BitBlt`(*dest_dc, dest_x, dest_y, width, height, src_dc, src_x, src_y, rop*)
Wraps the `BitBlt()` function. The C signature of `BitBlt()` is

```
BOOL BitBlt(HDC dest_dc, int dest_x, int dest_y, int width, int height, HDC src_dc, int src_x, int s
```

The function is called like this:

```
ret = BitBlt((HDC)dest_dc, dest_x, dest_y, width, height, (HDC)src_dc, src_x, src_y, (DWORD)rop);
```

Returns ret.

`CreateCaret`(*wnd, bitmap, width, height*)
Wraps the `CreateCaret()` function. The C signature of `CreateCaret()` is

```
BOOL CreateCaret(HWND wnd, HBITMAP bitmap, int width, int height)
```

The function is called like this:

```
ret = CreateCaret((HWND)wnd, (HBITMAP)bitmap, width, height);
```

Returns ret.

`CreateCompatibleBitmap`(*dc, width, height*)
Wraps the `CreateCompatibleBitmap()` function. The C signature of `CreateCompatibleBitmap()` is

```
HBITMAP CreateCompatibleBitmap(HDC dc, int width, int height)
```

The function is called like this:

```
ret = CreateCompatibleBitmap((HDC)dc, width, height);
```

Returns ret.

`CreateCompatibleDC`(*dc*)
Wraps the `CreateCompatibleDC()` function. The C signature of `CreateCompatibleDC()` is

```
HDC CreateCompatibleDC(HDC dc)
```

The function is called like this:

```
ret = CreateCompatibleDC((HDC)dc);
```

Returns ret.

`CreateDC`(*driver, device*)
Wraps the `CreateDC()` function. The C signature of `CreateDC()` is

```
HDC CreateDC(LPCSTR driver, LPCSTR device, LPCSTR output, DEVMODE *init)
```

The function is called like this:

`SetWindowLong`(*wnd, index, new_long*)
Wraps the `SetWindowLong()` function. The C signature of `SetWindowLong()` is

```
LONGOBJ SetWindowLong(HWND wnd, int index, LONG new_long)
```

The function is called like this:

```
ret = wnd_proc_setwindowlong(wnd, index, new_long);
```

Returns ret.

`SetWindowPos`(*wnd, wnd_insertafter, x, y, cx, cy, flags*)
Wraps the `SetWindowPos()` function. The C signature of `SetWindowPos()` is

```
BOOL SetWindowPos(HWND wnd, HWND wnd_insertafter, int x, int y, int cx, int cy, UINT flags)
```

The function is called like this:

```
ret = SetWindowPos((HWND)wnd, (HWND)wnd_insertafter, x, y, cx, cy, (UINT)flags);
```

Returns ret.

`SetWindowText`(*wnd, text*)
Wraps the `SetWindowText()` function. The C signature of `SetWindowText()` is

```
BOOL SetWindowText(HWND wnd, LPCSTR text)
```

The function is called like this:

```
ret = SetWindowText((HWND)wnd, (LPCSTR)text);
```

Returns ret.

`ShowCaret`(*wnd*)
Wraps the `ShowCaret()` function. The C signature of `ShowCaret()` is

```
BOOL ShowCaret(HWND wnd)
```

The function is called like this:

```
ret = ShowCaret((HWND)wnd);
```

Returns ret.

`ShowCursor`(*show*)
Wraps the `ShowCursor()` function. The C signature of `ShowCursor()` is

```
int ShowCursor(BOOL show)
```

The function is called like this:

```
ret = ShowCursor((BOOL)show);
```

Returns ret.

`ShowWindow`(*wnd, cmd_show*)

Wraps the `ShowWindow()` function. The C signature of `ShowWindow()` is

```
BOOL ShowWindow(HWND wnd, int cmd_show)
```

The function is called like this:

```
ret = ShowWindow((HWND)wnd, cmd_show);
```

Returns ret.

`TrackPopupMenu`(*menu, flags, x, y, wnd*)

Wraps the `TrackPopupMenu()` function. The C signature of `TrackPopupMenu()` is

```
BOOL TrackPopupMenu(HMENU menu, UINT flags, int x, int y, int reserved, HWND wnd, RECT *rect)
```

The function is called like this:

```
ret = TrackPopupMenu((HMENU)menu, (UINT)flags, x, y, 0, (HWND)wnd, NULL);
```

Returns ret.

`TranslateAccelerator`(*wnd, acc_table, msg*)

Wraps the `TranslateAccelerator()` function. The C signature of `TranslateAccelerator()` is

```
int TranslateAccelerator(HWND wnd, HACCEL acc_table, MSG *msg)
```

The function is called like this:

```
ret = TranslateAccelerator((HWND)wnd, (HACCEL)acc_table, &msg->msg);
```

Returns ret.

`TranslateMessage`(*msg*)

Wraps the `TranslateMessage()` function. The C signature of `TranslateMessage()` is

```
BOOL TranslateMessage(MSG *msg)
```

The function is called like this:

```
ret = TranslateMessage(&msg->msg);
```

Returns ret.

`UnregisterClass`(*name, inst*)

Wraps the `UnregisterClass()` function. The C signature of `UnregisterClass()` is

```
BOOL UnregisterClass(LPCSTR name, HINSTANCE inst)
```

The function is called like this:

```
ret = UnregisterClass((LPCSTR)name, (HINSTANCE)inst);
```

Returns ret.

`UpdateWindow`(*wnd*)

Wraps the `UpdateWindow()` function. The C signature of `UpdateWindow()` is

```
BOOL UpdateWindow(HWND wnd)
```

The function is called like this:

```
ret = UpdateWindow((HWND)wnd);
```

Returns ret.

`ValidateRect`(*wnd, rect*)

Wraps the `ValidateRect()` function. The C signature of `ValidateRect()` is

```
BOOL ValidateRect(HWND wnd, RECT *rect)
```

The function is called like this:

```
ret = ValidateRect((HWND)wnd, rect);
```

Returns ret.

`WritePrivateProfileString`(*app, key, val, file*)

Wraps the `WritePrivateProfileString()` function. The C signature of `WritePrivateProfileString()` is

```
BOOL WritePrivateProfileString(LPCSTR app, LPCSTR key, LPCSTR val, LPCSTR file)
```

The function is called like this:

```
ret = WritePrivateProfileString((LPCSTR)app, (LPCSTR)key, (LPCSTR)val, (LPCSTR)file);
```

Returns ret.

`Arc`(*dc, left, top, right, bottom, x_start, y_start, x_end, y_end*)

Wraps the `Arc()` function. The C signature of `Arc()` is

```
BOOL Arc(HDC dc, int left, int top, int right, int bottom, int x_start, int y_start, int x_end, int
```

The function is called like this:

```
ret = Arc((HDC)dc, left, top, right, bottom, x_start, y_start, x_end, y_end);
```

Returns ret.

`BeginPaint`(*wnd, ps*)

Wraps the `BeginPaint()` function. The C signature of `BeginPaint()` is

```
HDC BeginPaint(HWND wnd, PAINTSTRUCT *ps)
```

The function is called like this: