From billg Wed Feb 7 19:24:14 1990
To: nathanm
Subject: Funny calls
Date: Wed Feb 7 19:21:10 1990
Mail-Flags: 8000

Thought you might get a kick out of this.

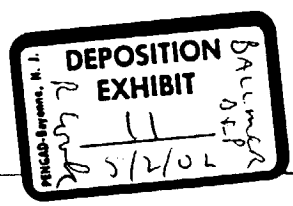
From mikemap Wed Feb 7 09:58:44 1990
To: billg steveb
Subject: Re: Message
Date: Wed Feb 07 09:53:56 1990

This looks right to me.

> >From steveb Tue Jan 30 17:15:08 1990
> To: billg mikemap
> Subject: Message
> Date: Tue Jan 30 17:15:06 1990
>
> does this look sane??

> From markwa Tue Jan 30 17:12:37 1990
> To: brianmac chrism chrisp leno steveb tomru
> Cc: bobga bobgu chasst jeffr jodys mikemap peteh philba vijayv
> Subject: Resolution of undocumented windows calls
> Date: Tue Jan 30 17:43:14 1990
>
> This email is a response to SteveB's request to publish undocumented Windows APIs used by MS applications, in order to avoid complaints from ISVs.

X0196134



The Windows group proposes resolution of the undocumented windows calls as summarized below. The list below of unpublished APIs includes all APIs identified by DavidWo that are called from at least one MS Windows application.

The Windows group proposes that calls to some of the undocumented APIs be removed from MS applications as soon as possible. These APIs should remain undocumented. The reason of leaving each of these APIs undocumented is provided at the end of this email.

Each Windows application group should confirm that will remove the undocumented API calls from its application at the earliest possible future release of its product.

Thanks

Unpublished API	Opus	Excl	Omga	Proj	PwrP	Status
AllocSelector	---	---	---	---	Yes	Will document in SDK
BeginDeferWindowPos	---	Yes	---	Yes	---	Will document in SDK
DeferWindowPos	---	Yes	---	Yes	---	Will document in SDK
DefineHandleTable	---	Yes	Yes	Yes	---	Will document in SDK
EndDeferWindowPos	---	Yes	---	Yes	---	Will document in SDK
EndMenu	Yes	Yes	Yes	Yes	---	Remove call from app
ExitWindows	Yes	Yes	Yes	Yes	---	Will document in SDK
FillWindow	Yes	---	Yes	---	---	Remove call from app
FreeSelector	---	---	---	---	Yes	Will document in SDK
Get80x87SaveSize	---	Yes	---	Yes	---	Remove call from app
GetCodeInfo	Yes	Yes	Yes	Yes	---	Will document in SDK
GetControlBrush	Yes	---	Yes	---	---	Remove call from app
GetCurPID	Yes	---	---	---	---	Remove call from app
GetCurrentPDB	Yes	Yes	Yes	---	---	Will document in SDK
GetPhysicalFontHandle	Yes	Yes	Yes	---	---	Remove call from app
GetRgnBox	---	Yes	Yes	---	---	Will document in SDK
GlobalDiscard	---	---	---	---	Yes	Remove call from app
InitApp	Yes	Yes	Yes	Yes	---	No need to document
InitTask	Yes	Yes	Yes	Yes	---	No need to document
KillSystemTimer	Yes	---	Yes	---	---	Remove call from app
LoadCursorIconHandler	Yes	---	---	---	---	Remove call from app
LongPtrAdd	---	---	---	---	Yes	Will document in SDK
MenuWndProc	Yes	---	---	---	---	Remove call from app
MulDiv	---	Yes	Yes	---	---	Will document in SDK
PatchCodeHandle	---	Yes	---	---	---	Remove call from app
SetSystemTimer	Yes	---	Yes	---	---	Remove call from app
ToAscii	---	Yes	Yes	---	---	Will document in SDK
WaitEvent	Yes	Yes	Yes	Yes	---	No need to document
__AHINCR	---	Yes	Yes	Yes	---	Already documented
API Count	15	18	17	12	4	

>From bobgu Fri Jan 26 14:12:02 1990

API's that will remain undocumented (and reasons why)

λ0196135

Get80x87SaveSize

We are adding a new bit to GetWinFlags to allow apps to determine

the existance of the 80x87 chip.

GetPhysicalFontHandle
This was only used to lock font bits down low. This has no meaning for pmode. Since the future is pmode there is no need to doc this.

GetCurPID
The info needed here can now be obtained with GetWinFlags().

GlobalDiscard
We can't seem to find this function anywhere. The orignal list stated that PwrP used this. Strange...

LoadCursorIconHandler
This function only handles 2.x icons/cursors. Also, 3.0 .DLL's have the resource loader set automatically.

MenuWndProc
Apps MUST not rely on the internal operation and message sequencing of the menu system.

PatchCodeHandle
This is only used by the Excel app loader. We want this to go away.

EndMenu
Apps were calling this to cancel menu mode. For 3.0 the message WM_CANCELMODE will do this.

GetControlBrush
All this function does is send a WM_CTLCOLOR message (documented) to the parent of the window specified. An app can, and should, do this directly.

FillWindow
All this function does is encapsulate sending a WM_CTLCOLOR message and doing a FillRect(). This function does nothing that an app cannot do with documented API/messages.

SetSystemTimer
KillSystemTimer
The system timer is reserved for use by Windows. It is fundamental to the proper operation of scroll bars, list boxes, carets, etc. Also, it appears that these really arn't used by the apps listed in the original mail (at least not in OMEGA).

...

InitApp (In the C startup code - need to doc for non-C apps)
InitTask (ditto)
WaitEvent (ditto)
__AHINCR (ditto)

InitApp, InitTask, WaitEven, and __AHINCR are called from startup code provided in xLIBCyW.LIB and xDLLCyW.LIB (or xWINLIBC.LIB until you start using the new Windows C run-time libraries and Windows import library LIBW.LIB).

All C apps and DLLs, as well as non-C (ASM) apps and DLLs, should be linked with xLIBCyW.LIB or xDLLCyW.LIB to pull in the startup code, even if the app or DLL doesn't make

any C run-time calls. If the app or DLL doesn't make C run-time calls, and/or doesn't need command line arguments or environment variables, you can stub out portions of the startup code by using OBJs provided in the SDK.

Since InitApp, InitTask, and WaitEvent are calls made from the startup code rather than directly from the application or DLL, there is no need to document these APIs.

__AHINCR is documented in the Programmers Guide chapter on advanced memory.