```
//************************************************************
//*       Macro:     Expert.wpl
//*     Purpose:     Expert macro library file
//*
//*      Author:     Paul Laing - WordPerfect Corporation
//*    Creation:     June 23, 1994
//************************************************************

// Revision History:
//
//   Date            Name                  Description of change

//************************************************
//   FUNCTION: GetQuickTasksDirectory
//   INPUTS: none
//   OUTPUTS: FilePath
//   USES: Wut1GetShared;CheckBackslash
//   DESC:
//************************************************
FUNCTION GetQuickTasksDirectory()
    KeyPath:="WPConfig"
    SubKey:="PRVQuickTasks"
    FilePath:=""
    DllCall(WPEXLink;"QueryReg";Result:INTEGER;
        {Address(KeyPath);Address(SubKey);Address(FilePath)})
    If(FilePath = "")
        FilePath:=WutlGetSharedPath()
    Endif
    FilePath:=CheckBackslash(FilePath)
    RETURN(FilePath)
ENDFUNC

//************************************************
//   FUNCTION: AddButton
//   DESC:
//************************************************
FUNCTION AddButton(LibraryPath;BitmapFile;TaskFile;BttnText;DescText;YellowText)
    Result:=AddButtonWithBMP(LibraryPath;BitmapFile;TaskFile;BttnText;DescText;
    YellowText;"";0)
    RETURN(Result)
ENDFUNC

//************************************************
//   FUNCTION: AddButtonWithBMP
//   DESC:
//************************************************
```

```
FUNCTION AddButtonWithBMP(LibraryPath;BitmapFile;TaskFile;BttnText;DescText;
YellowText;BitmapDLL;BitmapID)
    Result:=0 Bars:=""
    DllLoad(DADLink;LibraryPath+"DADEXT.DLL")
    DllCall(DADLink;"DoesThetaExist";ThetaResult:INTEGER;{WPString(DescText)})
    If (ThetaResult = 1)        // 11-10-94 Paull, changed return from BOOL to INTEGER
        MessageBox(Status:Result;Caption:"QuickTask";
        Message:"A QuickTask named """+DescText+""" already exists.  Do you wish to replace
        it?"; Style:YesNo!+IconQuestion!)
        If(Result = 7)
            DllFree(DADLink)
            Result:=0
            RETURN(Result)
        Endif
    Endif
    If(BitmapFile <> "")
        If(NOT DoesFileExist(Filename:BitmapFile))
            BitmapFile:=""
        Endif
    Endif
    If(StrLen(BttnText) > 47)
        BttnText:=SubStr(BttnText;1;47)
    Endif
    If(StrLen(DescText) > 47)
        DescText:=SubStr(DescText;1;47)
    Endif
    If(StrLen(TaskFile) > 256)
        TaskFile:=SubStr(TaskFile;1;47)
        MessageBox(Caption:"QuickTask";Message:"Pathname "+ToUpper(TaskFile)+". . .  is
        too long.";OK!+IconInformation!)
        DllFree(DADLink)
        Result:=0
        RETURN(Result)
    Endif
    If(DoesFileExist(Filename:TaskFile))
        BIFRead(
            Status:StatusVar;
            Group:"WordPerfect";
            Section:"Files";
            Item:"Default Template Extension";
            Value:TemplateExt;
            ExpectedItemType:AnsiString!;
            NameListType:Private!
        )
        If(StatusVar <= 0)
            TemplateExt:=".wpt"
```

```
Else
    TemplateExt:="."+TemplateExt
Endif

OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
WfsPathSplit(TaskFile;&OutDrive;&OutPathname;&OutFile;&OutExt)
FileType:=1
If(OutExt = TemplateExt or OutExt = ".exe" or OutExt = ".com")
    FileType:=0
Endif

Result:=1                                                    // True until False
If(DadChkBx1 = 1)                                            // Add button to Dad
    If(BitmapDll <> "" and BitmapFile = "")
        DllCall(DADLink;"SetBitmap";Result:Bool;
            {AnsiString(BitmapDLL);LoWord(BitmapID)})
    Endif
    DllCall(DADLink;"AddTask";Result:Bool;
        {AnsiString(BttnText);AnsiString(DescText);LoWord(2);
        AnsiString("");AnsiString(BitmapFile);
        WPString(YellowText);AnsiString(TaskFile);
        LoWord(FileType)})                          // 1=Macro type file, 0=Executable
    If(Result = False)
        MessageBox(Caption:"QuickTask";Message:"Adding button to DAD
        failed.";OK!+IconInformation!)
        DllFree(DADLink)
        Result:=0
        RETURN(Result)
    Endif
    Bars:="DAD"
Endif
If(DadChkBx2 = 1 and Result <> 0)          // Add button to Quick Tasks
    If(BitmapDll <> "" and BitmapFile = "")
        DllCall(DADLink;"SetBitmap";Result:Bool;
            {AnsiString(BitmapDLL);LoWord(BitmapID)})
    Endif
    DllCall(DADLink;"AddTask";Result:Bool;
        {AnsiString(BttnText);AnsiString(DescText);LoWord(1);
        AnsiString("");AnsiString(BitmapFile);
        WPString(YellowText);AnsiString(TaskFile);
        LoWord(FileType)})                          // 1=Macro type file, 0=Executable

    If(Result = False)
        MessageBox(Caption:"QuickTask";Message:"Adding button to QuickTasks
        failed.";OK!+IconInformation!)
        DllFree(DADLink)
```

```
                Result:=0
                RETURN(Result)
            Endif
            If(Bars = "DAD")
                Bars:=Bars+" and the QuickTasks list"
            Else
                Bars:="the QuickTasks list"
            Endif
        Endif
    Else
        MessageBox(Caption:"File Not Found";
        Message:"File "+ToUpper(TaskFile)+" not found.";Style:OK!+IconInformation!)
    Endif
    If(Bars <> "")
        MessageBox(Caption:"QuickTask";
        Message:"Your personal QuickTask has been added to "+Bars+".";Style:OK!
        +IconInformation!)
    Endif
    DllFree(DADLink)
    RETURN(Result)
ENDFUNC

//************************************************
//  FUNCTION: CreateFile
//  DESC:
//************************************************
FUNCTION CreateFile(PathFilename)
    FileID:=OpenFile(
        PathFilename;
        AccessMode:WriteNew!;
        ShareMode:Exclusive!;
        DataType:OEMText!
    )
    RETURN(FileID)
ENDFUNC

//************************************************
//  FUNCTION: AddApp
//  DESC:
//************************************************
PROCEDURE AddApp(FileID;AppID)
    Switch(AppID)
    CaseOf 2:
        FileWrite(FileID:FileID;
        Data:"Application(A2;""WordPerfect"";""UK"")";
        NewLine:NewLine!)
```

```
    CaseOf 3:
        FileWrite(FileID:FileID;
        Data:"Application(A3;""WPPrWin"";""UK"")";
        NewLine:NewLine!)
    CaseOf 4:
        FileWrite(FileID:FileID;
        Data:"Application(A4;""WPOffice"";""UK"")";
        NewLine:NewLine!)
    CaseOf 5:
        FileWrite(FileID:FileID;
        Data:"Application(A5;""QuattroPro"";""US"")";
        NewLine:NewLine!)
    EndSwitch
ENDPROC

//************************************************
//   PROCEDURE: WriteString
//   DESC:
//************************************************
PROCEDURE WriteString(FileID;MacroString)
    FileWrite(FileID:FileID;
        Data:MacroString;
        NewLine:NewLine!)
ENDPROC

//************************************************
//*  PROCEDURE: WriteStrings
//*  DESC: Write strings to an open file
//*  INPUTS: FileID;NumLines;&MacroStrings[]
//*  OUTPUTS: none
//************************************************
PROCEDURE WriteStrings(FileID;NumLines;&MacroStrings[])
    For(Incr; 1; Incr <= NumLines; Incr + 1)
        FileWrite(FileID:FileID;
            Data:MacroStrings[Incr];
            NewLine:NewLine!)
    EndFor
ENDPROC

//************************************************************
//*  FUNCTION: CheckBackslash
//*  DESC: Add a backslash to the end of a string if one is not found
//*  INPUT: InSpec
//*  OUTPUT: String with a backslash added if one is not found
//************************************************************
FUNCTION CheckBackslash(InSpec)
```

```
    DllCall(SHWINLink;"WfsAddPathSlash";Result:INTEGER;
        {Address(InSpec)})
    RETURN(InSpec)
ENDFUNC

//*************************************************************
//   FUNCTION: Pathname
//   DESC:
//   INPUT: InSpec
//   OUTPUT: Path name without filename
//   USES: WfsPathSplit
//*************************************************************
FUNCTION Pathname(InSpec)
    OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
    WfsPathSplit(InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt)
    Result:=OutDrive+OutPathname
    RETURN(Result)
ENDFUNC    // Pathname

//*************************************************************
//   FUNCTION: Filename
//   DESC:
//   INPUT: InSpec
//   OUTPUT: Filename
//   USES: WfsPathSplit
//*************************************************************
FUNCTION Filename(InSpec)
    OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
    WfsPathSplit(InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt)
    Result:=OutFile
    RETURN(Result)
ENDFUNC    // Filename

//*************************************************************
//   FUNCTION: Extension
//   DESC:
//   INPUT: InSpec
//   OUTPUT: Dot plus three letter extension
//   USES: WfsPathSplit
//*************************************************************
FUNCTION Extension(InSpec)
    OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
    WfsPathSplit(InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt)
    If(OutExt <> "")
        Len:=StrLen(OutExt)
        Result:=SubStr(OutExt;2;Len - 1)
```

```
    Endif
    RETURN(Result)
ENDFUNC     // Extension

//*****************************************************************
//*  PROCEDURE: WfsPathSplit
//*  DESC:
//*  INPUTS: InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt
//*  OUTPUTS: none
//*****************************************************************
PROCEDURE WfsPathSplit(InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt)
    DllCall(SHWINLink;"WfsPathSplit";:VOID;
        {Address(InSpec);Address(OutDrive);Address(OutPathname);
        Address(OutFile);Address(OutExt)})
ENDPROC

//*****************************************************************
//*  FUNCTION: WfsDoesExist
//*  DESC:
//*  INPUTS: InSpec;Replace
//*  OUTPUTS: Result
//*  USES: WfsPathSplit
//*****************************************************************
FUNCTION WfsDoesExist(&InSpec;InDefault;ReplacePrompt)
    If(InDefault <> "")
        OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
        WfsPathSplit(InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt)
        If(OutPathname = "")
            InSpec:=InDefault+OutFile+OutExt
        Endif
    Endif

    OutPath:=""
    OutName:=""
    DllCall(SHWINLink;"WfsDoesExist";Result:Word;
        {Address(InSpec);Address(OutPath);Address(OutName)})
    Switch(Result)
    CaseOf 101:
        If(ReplacePrompt)
            MessageBox(Status:Result;Caption:"Replace";
            Message:"Replace "+ToUpper(InSpec)+"?";
            Style:YesNo!+IconQuestion!)
            If(Result = 6)
                Result:=201
            Else
                Result:=101
```

```
            Endif
        Endif

    CaseOf 102:
        MessageBox(Caption:"Filename Error";
        Message:"No filename in: "+ToUpper(InSpec);
        Style:OK!+IconInformation!)
        Result:=102

    CaseOf 103:
        Result:=103

    CaseOf 104;109:
        MessageBox(Caption:"Filename Error";
        Message:"Invalid filename: "+ToUpper(InSpec);
        Style:OK!+IconInformation!)
        Result:=109

    CaseOf 110:
        OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
        WfsPathSplit(InSpec;&OutDrive;&OutPathname;&OutFile;&OutExt)
        If(StrPos(OutPathname;" ") = 0)
            MessageBox(Status:Result;Caption:"Create Directory";
            Message:"Directory "+ToUpper(OutDrive)+ToUpper(OutPathname)+" does not exist.
            Do you wish to create the directory?";Style:YesNo!+IconQuestion!)
            If(Result = 6)
                CreateDirectory(DirectoryName:OutDrive+OutPathname;Prompts:NoPrompts!)
                InSpec:=OutDrive+OutPathname+OutFile+OutExt
                Result:=210
            Else
                Result:=110
            Endif
        Else
            MessageBox(Caption:"Path Error";
            Message:"Invalid path: "+ToUpper(OutDrive)+ToUpper(OutPathname);
            Style:OK!+IconInformation!)
            Result:=110
        Endif

    Default:
        Result:=-1

    EndSwitch
    RETURN(Result)
ENDFUNC
```

```
//*****************************************************************
//*  FUNCTION: WriteBIF
//*  DESC: Writes items to the private Bif file
//*  INPUTS: Section;Item;TextIn
//*  OUTPUTS: StatusVar
//*****************************************************************
FUNCTION WriteBIF(Section;Item;TextIn)
    BIFWrite(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:Section;
        Item:Item;
        Value:TextIn;
        ItemType:AnsiString!;
        ItemFlags:1
    )
    RETURN(StatusVar)
ENDFUNC

//*****************************************************************
//*  FUNCTION: ReadBIF
//*  DESC: Reads items from the private Bif
//*  INPUTS: Section;Item
//*  OUTPUTS: TextOut
//*****************************************************************
FUNCTION ReadBIF(Section;Item)
    BIFRead(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:Section;
        Item:Item;
        Value:TextOut;
        ExpectedItemType:AnsiString!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        RETURN(TextOut)
    Else
        RETURN("")
    Endif
ENDFUNC

//*****************************************************************
//*  SUBROUTINE: WriteBIFFromList
//*  DESC: Writes items from a list item control to the private Bif file
//*  INPUTS: hWnd;BoxType;Section;Item
```

```
//*  OUTPUTS: none
//*****************************************************************
PROCEDURE WriteBIFFromList(hWnd;BoxType;Section;Item)
    If(BoxType = 1)
        FoundDuplicate:=False
        TextIn:=WMGetText(hWnd)
        NumItems:=CBGetCount(hWnd)
        For(Incr; 1; Incr <= NumItems; Incr + 1)
            BIFRead(
                Status:StatusVar;
                Group:"WPIntegrator";
                Section:Section;
                Item:Item+" "+Incr;
                Value:TextOut;
                ExpectedItemType:AnsiString!;
                NameListType:Private!
            )
            If(StatusVar > 0)
                If(TextIn = TextOut)
                    FoundDuplicate:=True
                Endif
            Endif
        Endfor
        If(NOT FoundDuplicate)
            CBAddString(hWnd;TextIn)
            NumItems:=CBGetCount(hWnd)
        Endif
    Else
        NumItems:=LBGetCount(hWnd)
    Endif
    BIFWrite(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:Section;
        ItemType:AnsiString!;
        ItemFlags:1
    )
    For(Incr; 1; Incr <= NumItems; Incr + 1)
        If(BoxType = 1)
            TextOut:=CBGetLBText(hWnd;Incr - 1)
        Else
            TextOut:=LBGetText(hWnd;Incr - 1)
        Endif
        BIFWrite(
            Status:StatusVar;
            Group:"WPIntegrator";
```

```
            Section:Section;
            Item:Item+" "+Incr;
            Value:TextOut;
            ItemType:AnsiString!;
            ItemFlags:1
        )
    Endfor
ENDPROC

//*****************************************************************
//*  SUBROUTINE: ReadBIFToList
//*  DESC: Reads items from the private Bif file into a list item control
//*  INPUTS: hWnd;BoxType;Section;Item
//*  OUTPUTS: none
//*****************************************************************
PROCEDURE ReadBIFToList(hWnd;BoxType;Section;Item)
    BIFInfo(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:Section;
        UseCount:NumItems;
        ItemInfoType:Private!
    )
    If(StatusVar = 0)
        For(Incr; 1; Incr <= NumItems; Incr + 1)
            BIFRead(
                Status:StatusVar;
                Group:"WPIntegrator";
                Section:Section;
                Item:Item+" "+Incr;
                Value:TextIn;
                ExpectedItemType:AnsiString!;
                NameListType:Private!
            )
            If(StatusVar > 0)
                If(BoxType = 1)
                    CBInsertString(hWnd;Incr - 1;TextIn)
                Else
                    LBInsertString(hWnd;Incr - 1;TextIn)
                Endif
            Endif
        Endfor
    Endif
ENDPROC

//*****************************************************************
```

```
//*  FUNCTION: ToggleDocSelDialog
//*  DESC: Toggle the state of the document select dialog in Presentations
//*  INPUTS: State
//*****************************************************************
FUNCTION ToggleDocSelDialog(State)
    Result:=0
    BIFRead(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:"WPPrWin";
        Item:"Use DocSel Dialog";
        Value:ValueOut;
        ExpectedItemType:UnsignedWord!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        BIFWrite(
            Status:StatusVar;
            Group:"WPPrWin";
            Section:"Environment Settings";
            Item:"Use DocSel Dialog";
            Value:ValueOut;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
        Result:=1
    Endif
    BIFRead(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:"WPPrWin";
        Item:"Default Document Type";
        Value:ValueOut;
        ExpectedItemType:UnsignedWord!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        BIFWrite(
            Status:StatusVar;
            Group:"WPPrWin";
            Section:"Environment Settings";
            Item:"Default Document Type";
            Value:ValueOut;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
```

```
Endif
BIFWrite(
    Status:StatusVar;
    Group:"WPIntegrator";
    Section:"WPPrWin";
    ItemType:UnsignedWord!;
    ItemFlags:1
)
If(NOT State)
    BIFRead(
        Status:StatusVar;
        Group:"WPPrWin";
        Section:"Environment Settings";
        Item:"Use DocSel Dialog";
        Value:ValueOut;
        ExpectedItemType:UnsignedWord!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        BIFWrite(
            Status:StatusVar;
            Group:"WPIntegrator";
            Section:"WPPrWin";
            Item:"Use DocSel Dialog";
            Value:ValueOut;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
        If(ValueOut = 1)
            ValueIn:=0
            BIFWrite(
                Status:StatusVar;
                Group:"WPPrWin";
                Section:"Environment Settings";
                Item:"Use DocSel Dialog";
                Value:ValueIn;
                ItemType:UnsignedWord!;
                ItemFlags:1
            )
            Result:=1
        Endif
    Else
        ValueIn:=0
        BIFWrite(
            Status:StatusVar;
            Group:"WPIntegrator";
```

```
            Section:"WPPrWin";
            Item:"Use DocSel Dialog";
            Value:ValueIn;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
        BIFWrite(
            Status:StatusVar;
            Group:"WPPrWin";
            Section:"Environment Settings";
            Item:"Use DocSel Dialog";
            Value:ValueIn;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
        Result:=1
    Endif
    BIFRead(
        Status:StatusVar;
        Group:"WPPrWin";
        Section:"Environment Settings";
        Item:"Default Document Type";
        Value:ValueOut;
        ExpectedItemType:UnsignedWord!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        BIFWrite(
            Status:StatusVar;
            Group:"WPIntegrator";
            Section:"WPPrWin";
            Item:"Default Document Type";
            Value:ValueOut;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
        ValueIn:=68             // Drawing document type
        BIFWrite(
            Status:StatusVar;
            Group:"WPPrWin";
            Section:"Environment Settings";
            Item:"Default Document Type";
            Value:ValueIn;
            ItemType:UnsignedWord!;
            ItemFlags:1
        )
```

```
        Else
            ValueIn:=68             // Drawing document type
            BIFWrite(
                Status:StatusVar;
                Group:"WPIntegrator";
                Section:"WPPrWin";
                Item:"Default Document Type";
                Value:ValueIn;
                ItemType:UnsignedWord!;
                ItemFlags:1
            )
            BIFWrite(
                Status:StatusVar;
                Group:"WPPrWin";
                Section:"Environment Settings";
                Item:"Default Document Type";
                Value:ValueIn;
                ItemType:UnsignedWord!;
                ItemFlags:1
            )
        Endif
    Endif
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: ToggleStartupCoach
//*  DESC: Toggle the state of a product start up Coach
//*  INPUTS: State
//*****************************************************************
FUNCTION ToggleStartupCoach(State;AppID)
    Switch(AppID)
    CaseOf 2:
        Product:="WordPerfect"
    CaseOf 3:
        Product:="WPPrWin"
    Default:
        Result:=-2
        RETURN(Result)
    EndSwitch
    Result:=0
    BIFRead(
        Status:StatusVar;
        Group:"WPIntegrator";
        Section:Product;
        Item:"QuickStart Coach";
```

```
        Value:ValueOut;
        ExpectedItemType:Boolean!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        BIFWrite(
            Status:StatusVar;
            Group:Product;
            Section:"Coaches";
            Item:"QuickStart";
            Value:ValueOut;
            ItemType:Boolean!;
            ItemFlags:1
        )
        Result:=1
        BIFWrite(
            Status:StatusVar;
            Group:"WPIntegrator";
            Section:Product;
            ItemType:Boolean!;
            ItemFlags:1
        )
    Endif
    If(NOT State)
        BIFRead(
            Status:StatusVar;
            Group:Product;
            Section:"Coaches";
            Item:"QuickStart";
            Value:ValueOut;
            ExpectedItemType:Boolean!;
            NameListType:Private!
        )
        If(StatusVar > 0)
            BIFWrite(
                Status:StatusVar;
                Group:"WPIntegrator";
                Section:Product;
                Item:"QuickStart Coach";
                Value:ValueOut;
                ItemType:Boolean!;
                ItemFlags:1
            )
            If(ValueOut = False)
                ValueIn:=True
                BIFWrite(
```

```
                    Status:StatusVar;
                    Group:Product;
                    Section:"Coaches";
                    Item:"QuickStart";
                    Value:ValueIn;
                    ItemType:Boolean!;
                    ItemFlags:1
                )
                Result:=1
            Endif
        Else
            ValueIn:=False
            BIFWrite(
                Status:StatusVar;
                Group:"WPIntegrator";
                Section:Product;
                Item:"QuickStart Coach";
                Value:ValueIn;
                ItemType:Boolean!;
                ItemFlags:1
            )
            BIFWrite(
                Status:StatusVar;
                Group:Product;
                Section:"Coaches";
                Item:"QuickStart";
                Value:ValueIn;
                ItemType:Boolean!;
                ItemFlags:1
            )
            Result:=1
        Endif
    Endif
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  PROCEDURE: ExpertInit
//*  DESC: Initializes an Expert
//*  INPUTS: none
//*  OUTPUTS: none
//*************************************************************
PROCEDURE ExpertInit()
    GLOBAL(TitleName)
    GLOBAL(USERLink;WINLink;SHWINLink;WPEXLink;STEXLink)
    GLOBAL(hWndDescription;hWndHintText;hWndBmp;hWndFrame)
```

```
    GLOBAL(hWndPrev;hWndNext;hWndHint;HelpFile)
    GLOBAL(hMemory;LPMemory)
    GLOBAL(TextIn;TextOut) TextIn:="" TextOut:=""
    GLOBAL(LibraryPath)
    GLOBAL(hWndBannerBmp)
    GLOBAL(Enabled) Enabled:=False
    GLOBAL(FinishBttn) FinishBttn:=True
    GLOBAL(CancelString) CancelString:="QuickTask"
ENDPROC

//*************************************************************
//*  PROCEDURE: WaitPrompt
//*  DESC: Toggles wait prompt dialog
//*  INPUTS: State;TextIn
//*  OUTPUTS:  none
//*************************************************************
PROCEDURE WaitPrompt(State;TextIn)
    If(State)
        If(Exists(WaitDlgExists) = 0)
            GLOBAL(WaitDlgExists)
            WaitDlgExists:=True

            If(Exists(WaitDlgID) = 0)
                PERSIST(WaitDlgID)
                WaitDlgID:=1
            Else
                WaitDlgID:=WaitDlgID + 1
            Endif
            DialogDefine("WaitDlg" + WaitDlgID;50;50;187;43;Cancel!+Percent!+NoTitle!;"")
            RegionShowWindow("WaitDlg"+WaitDlgID+".CancelBttn";Hide!)
            DialogAddText("WaitDlg" + WaitDlgID;"T1";8;8;60;10;1;"Please Wait . . .")
            DialogAddText("WaitDlg" + WaitDlgID;"T2";8;19;155;10;1;TextIn)
        Endif
        DialogDisplay("WaitDlg" + WaitDlgID;2;WaitMessageCB)
    Else
        DialogUndisplay("WaitDlg" + WaitDlgID;2)
    Endif
ENDPROC

PROCEDURE WaitMessageCB()
    If(WaitMessageCB[3] = "CancelBttn")
        If(Exists(CancelString) = 0)
            GLOBAL(CancelString)
            CancelString:="QuickTask"
        Endif
        MessageBox(Status:Result;Caption:"Cancel";
```

```
        Message:"Cancel "+CancelString+"?";Style:YesNo!+IconQuestion!)
        If(Result = 6)
            If(Exists(hMemory) <> 0)
                GlobalUnlock(hMemory)
                GlobalFree(hMemory)
            Endif
            QUIT
        Endif
    Endif
ENDPROC

//************************************************
//  PROCEDURE: ExpertPrmptIns
//  DESC:
//************************************************
PROCEDURE ExpertPrmptIns(Title;Text;BttnText;hWndParent)
    DllLoad(hLink;"GDI.EXE")

    DllCall(USERLink;"GetSystemMetrics";Result:Word;{LoWord(0)})
    FontSize:=12
    If(Result = 1024)
        FontSize:=14
    Endif

    DialogDefine(Dialog:"ExpertPrmptIns";
        Left:100;Top:0;Width:80;Height:60;
        Style:Percent!+Modeless!;Caption:Title)
    //DialogHandle(hWndChild;"ExpertPrmptIns")

    If(BttnText = "")
        BttnText:="&Next >"
    Endif
    DialogAddPushButton("ExpertPrmptIns";"PromptBttn";
        18;27;40;13;
        1;BttnText)

    DialogAddText(Dialog:"ExpertPrmptIns";Control:"Description";
        Left:6;Top:3;Width:65;Height:25;
        Style:1;Text:Text)
    DialogHandle(hWnd;"ExpertPrmptIns";"Description")

    DllCall(hLink;"CreateFont";hFont:Word;
        {LoWord(FontSize);LoWord(0);LoWord(0);LoWord(0);
        LoWord(0);LoWord(0);LoWord(0);LoWord(0);LoWord(0);
        LoWord(0);LoWord(0);LoWord(0);LoWord(0);"Ariel"})
```

```
    DllCall(USERLink;"SendMessage";Result:DWord;
        {LoWord(hWnd);LoWord(48);LoWord(hFont);1})        // 48 = WM_SETFONT

    //SetParent(hWndChild;hWndParent)

    DialogDisplay("ExpertPrmptIns";1;ExpertPrmptInsCB)
    SetFocus(hWndParent)

    If(Exists(PromptDone) = 0)
        GLOBAL(PromptDone)
    Endif
    PromptDone:=False
    While(NOT PromptDone)
    EndWhile
    Discard(PromptDone)
    DialogDestroy("ExpertPrmptIns")
ENDPROC

PROCEDURE ExpertPrmptInsCB()
    If(ExpertPrmptInsCB[3] = "PromptBttn")
        PromptDone:=True
    Endif
ENDPROC

//************************************************
//   PROCEDURE: ExpertPrompt
//   DESC:
//************************************************
PROCEDURE ExpertPrompt(Title;Text;BttnText)
    DllLoad(hLink;"GDI.EXE")

    DllCall(USERLink;"GetSystemMetrics";Result:Word;{LoWord(0)})
    FontSize:=12
    If(Result = 1024)
        FontSize:=14
    Endif

    DialogDefine(Dialog:"ExpertPrompt";
        Left:100;Top:0;Width:80;Height:60;
        Style:Percent!+Modeless!;Caption:Title)

    If(BttnText = "")
        BttnText:="Next >"
    Endif
    DialogAddPushButton("ExpertPrompt";1;
        18;27;40;13;
```

```
        1;BttnText)

    DialogAddText(Dialog:"ExpertPrompt";Control:"Description";
        Left:6;Top:3;Width:65;Height:25;
        Style:1;Text:Text)

    DialogHandle(hWnd;"ExpertPrompt";"Description")

    DllCall(hLink;"CreateFont";hFont:Word;
        {LoWord(FontSize);LoWord(0);LoWord(0);LoWord(0);
        LoWord(0);LoWord(0);LoWord(0);LoWord(0);LoWord(0);
        LoWord(0);LoWord(0);LoWord(0);LoWord(0);"Ariel"})

    DllCall(USERLink;"SendMessage";Result:DWord;
        {LoWord(hWnd);LoWord(48);LoWord(hFont);1})        // 48 = WM_SETFONT

    DialogDisplay("ExpertPrompt";1)
    DialogDestroy("ExpertPrompt")
ENDPROC

//************************************************
//  PROCEDURE:  MainDlg
//  DESC:
//************************************************
PROCEDURE MainDlg(DialogID)
    QuickTaskDlg(DialogID;True;False)
ENDPROC

//************************************************
//  PROCEDURE:  QuickTaskDlg
//  DESC:
//************************************************
PROCEDURE QuickTaskDlg(DialogID;DefButton;HelpBttn)
    DialogDefine(Dialog:DialogID;
        Left:50;Top:50;Width:286;Height:188;
        Style:Cancel!+Modeless!+Percent!;Caption:TitleName)

    RegionShowWindow(NamedRgn:DialogID+".CancelBttn";State:Hide!)
    BttnDef:=0
    If(DefButton)
        BttnDef:=1
    Endif
    DialogAddPushButton(DialogID;307;
        226;151;50;13;
        BttnDef;"&Next >")
    DialogHandle(hWndNext;DialogID;307)
```

```
    DialogAddPushButton(DialogID;306;
        172;151;50;13;
        0;"< &Previous")
    DialogHandle(hWndPrev;DialogID;306)
    If(HelpBttn)
        DialogAddPushButton(DialogID;304;
            110;151;50;13;
            0;"&Help")
    Endif
    DialogAddPushButton(DialogID;302;
        57;151;50;13;
        0;"&Tip")
    DialogHandle(hWndHint;DialogID;302)
    DialogAddPushButton(DialogID;2;
        4;151;50;13;
        0;"Cancel")

    // Graphic Frame
    DialogAddFrame(Dialog:DialogID;Control:"GraphicFrame";
        Left:4;Top:4;Width:123;Height:140;
        Style:2)
//  DialogAddText(Dialog:DialogID;Control:"GraphicFrame";
//      Left:4;Top:4;Width:123;Height:140;
//      Style:16;Text:"")
    DialogHandle(hWndFrame;DialogID;"GraphicFrame")

    DialogAddControl(Dialog:DialogID;Control:12;
    0; 0; 0; 0;
    Class:"WPbmp20";Style:4+7+8;WindowName:"";MacroVar:Result;Instance:0)
    DialogHandle(hWndBannerBmp;DialogID;12)

    DialogAddControl(Dialog:DialogID;Control:13;
    0; 0; 0; 0;
    Class:"WPbmp20";Style:4+7+8;WindowName:"";MacroVar:Result;Instance:0)
    DialogHandle(hWndBmp;DialogID;13)

    // Text Frame
    DialogAddFrame(Dialog:DialogID;Control:"Frame";
        Left:131;Top:4;Width:145;Height:140;
        Style:2)
//  DialogAddText(Dialog:DialogID;Control:"Frame";
//      Left:131;Top:4;Width:145;Height:140;
//      Style:16;Text:"")

    DialogAddText(Dialog:DialogID;Control:"Description";
        Left:136;Top:9;Width:135;Height:130;
```

```
        Style:1;Text:"")
    DialogHandle(hWndDescription;DialogID;"Description")

    DialogAddText(Dialog:DialogID;Control:"HintText";
        Left:9;Top:9;Width:113;Height:130;
        Style:1;Text:"")
    DialogHandle(hWndHintText;DialogID;"HintText")
ENDPROC

//************************************************
//  PROCEDURE: DADScreenOne
//  DESC:
//************************************************
PROCEDURE DADScreenOne(DialogID;&Offset;LPMemory;FeatureID)
    GLOBAL(DadChkBx1;DadChkBx2)
    DadChkBx1:=0 DadChkBx2:=0
    LoadOffset:=Offset
    // Screen 1 Controls
    Stuff:=2                                    // number of controls for this set
    Stuff(&Stuff;LPMemory;Offset;2)

    DialogAddCheckBox(DialogID;"DadChkBx1";
        145;90;100;11;
        "Add as button on &DAD";Var)
    DialogHandle(Stuff;DialogID;"DadChkBx1")
    Offset:=Offset + 2
    Stuff(&Stuff;LPMemory;Offset;2)

    DialogAddCheckBox(DialogID;"DadChkBx2";
        145;100;100;11;
        "Add to &QuickTask list";Var)
    DialogHandle(Stuff;DialogID;"DadChkBx2")
    Offset:=Offset + 2
    Stuff(&Stuff;LPMemory;Offset;2)

    LoadExpertControls(FeatureID;LPMemory + LoadOffset)
    Offset:=Offset + 2
ENDPROC

//************************************************
//  PROCEDURE: DADScreenTwo
//  DESC:
//************************************************
PROCEDURE DADScreenTwo(DialogID;&Offset;LPMemory;FeatureID)
    LoadOffset:=Offset
    // Screen 2 Controls
```

```
    Stuff:=6                                    // number of controls for this set
    Stuff(&Stuff;LPMemory;Offset;2)

    DialogAddText(Dialog:DialogID;Control:"DadStatic1";
        Left:141;Top:76;Width:121;Height:10;
        Style:1;Text:"Tas&k name:")
    DialogHandle(Stuff;DialogID;"DadStatic1")
    Offset:=Offset + 2
    Stuff(&Stuff;LPMemory;Offset;2)
    DialogAddEditBox(Dialog:DialogID;Control:"DadEdBx1";
        Left:141;Top:87;Width:121;Height:13;
        Style:0;MacroVar:TaskDescr;LimitText:100)
    DialogHandle(Stuff;DialogID;"DadEdBx1")
    Offset:=Offset + 2
    Stuff(&Stuff;LPMemory;Offset;2)

    DialogAddText(Dialog:DialogID;Control:"DadStatic2";
        Left:141;Top:106;Width:80;Height:10;
        Style:1;Text:"&Filename:")
    DialogHandle(Stuff;DialogID;"DadStatic2")
    Offset:=Offset + 2
    Stuff(&Stuff;LPMemory;Offset;2)
    QuickTaskDir:=GetQuickTasksDirectory()
    DialogAddFileNameBox(Dialog:DialogID;Control:"DadFlnmEd";
        Left:140;Top:117;Width:121;Height:15;
        Style:FilesAndDirs!;MacroVar:Filename;DefaultDir:QuickTaskDir;Template:"*.wcm")
    DialogHandle(Stuff;DialogID;"DadFlnmEd")
    Offset:=Offset + 2
    Stuff(&Stuff;LPMemory;Offset;2)

    LoadExpertControls(FeatureID;LPMemory + LoadOffset)
    Offset:=Offset + 2
ENDPROC

//*************************************************************
//*  FUNCTION: GetWindowsDirectory
//*  DESC: Return the current directory where Windows is running from
//*************************************************************
FUNCTION GetWindowsDirectory()
    WinDir:=""
    DllCall(WINLink;"GetWindowsDirectory";Result:Word;
        {Address(WinDir);LoWord(256)})          // Get Windows directory
    WinDir:=ToLower(WinDir)+"\"
    RETURN(WinDir)
ENDFUNC
```

```
//*************************************************************
//*  FUNCTION:WutlGetSharedPath
//*  DESC: Get shared code path
//*************************************************************
FUNCTION WutlGetSharedPath()
    SharedPath:=""
    DllCall(SHWINLink;"WutlGetSharedPath";:VOID;
        {Address(AnsiString(SharedPath));LoWord(64)})
    RETURN(SharedPath)
ENDFUNC

//*************************************************************
//*  FUNCTION: ExpertInitializeDialog
//*  DESC: Initialize the Expert dialog with .DLL
//*************************************************************
FUNCTION ExpertInitializeDialog(hWndDescription;hWndHintText;hWndBmp;hWndFrame;
hWndPrev;hWndNext;hWndHint;FeatureID;HelpFile;hWndResource)
    DllCall(WPEXLink; "ExpertInitializeDialog"; Result:Bool;
        {hWndDescription;hWndHintText;hWndBmp;hWndFrame;
        hWndPrev;hWndNext;hWndHint;LoWord(FeatureID);
        AnsiString(HelpFile);hWndResource;LoWord(ExpertID)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: ExpertInitializeBanner
//*  DESC: Initialize the a banner bitmap on the Expert dialog
//*************************************************************
FUNCTION ExpertInitializeBanner(hWndBannerBmp)
    DllCall(WPEXLink; "ExpertInitializeBanner"; Result:Bool;
        {hWndBannerBmp})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: ExpertUpdateDialog
//*  DESC: Update macro dialog with new feature
//*************************************************************
FUNCTION ExpertUpdateDialog(FeatureID)
    DllCall(WPEXLink;"ExpertUpdateDialog";Result:Word;
        {LoWord(FeatureID);LoWord(ExpertID)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: FileType
```

```
//*  DESC: Update macro dialog with new feature
//*****************************************************************
FUNCTION FileType(Filename)
    DllCall(WPEXLink;"FileType";Result:Word;{Address(Filename)})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: ShFileType
//*  DESC: Update macro dialog with new feature
//*****************************************************************
FUNCTION ShFileType(Filename)
    DllCall(SHWINLink;"WcvtDetectFileFormat";Result:Word;
        {Address(Filename)})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  PROCEDURE: HintToggle
//*  DESC: Toggle hint with Expert bitmap or visa versa
//*****************************************************************
PROCEDURE HintToggle()
    DllCall(WPEXLink;"HintToggle";:VOID;{LoWord(ExpertID)})
ENDPROC

//*****************************************************************
//*  FUNCTION: LoadExpertControls
//*  DESC: Load all control hWnd values into the .DLL by feature
//*****************************************************************
FUNCTION LoadExpertControls(FeatureID;LPMemory)
    DllCall(WPEXLink;"LoadExpertControls";Result:Word;
        {LoWord(FeatureID);LPMemory;LoWord(ExpertID)})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: GlobalAlloc
//*  DESC: Create a memory structure.
//*****************************************************************
FUNCTION GlobalAlloc(BufferSize)
    DllCall(WINLink;"GlobalAlloc";hMemory:Word;
        {LoWord(8194);BufferSize}) // 8194 = 0x02002  Movable & sharable
    RETURN(hMemory)
ENDFUNC

//*****************************************************************
```

```
//*  FUNCTION: GlobalLock
//*  DESC: Lock down a memory structure.
//*****************************************************************
FUNCTION GlobalLock(hMemory)
    DllCall(WINLink;"GlobalLock";LPMemory:DWord;
        {LoWord(hMemory)})
    RETURN(LPMemory)
ENDFUNC

//*****************************************************************
//*  PROCEDURE: GlobalUnlock
//*  DESC: Unlock a locked down memory structure.
//*****************************************************************
PROCEDURE GlobalUnlock(hMemory)
    DllCall(WINLink;"GlobalUnLock";Junk:Word;
        {LoWord(hMemory)})
ENDPROC

//*****************************************************************
//*  PROCEDURE: Stuff
//*  DESC: Stuff an item into a memory structure.
//*****************************************************************
PROCEDURE Stuff(&Stuff;LPMemory;Offset;DataSize)
    DllCall(SHWINLink;"WmemMoveMemory";:VOID;
        {LPMemory+Offset;Address(Stuff);LoWord(DataSize)})
ENDPROC

//*****************************************************************
//*  FUNCTION: Pull
//*  DESC: Pull an item from a memory structure.
//*  INPUTS: &Pull;LPMemory;Offset;DataSize
//*  OUTPUTS: Pull
//*****************************************************************
FUNCTION Pull(&Pull;LPMemory;Offset;DataSize)
    DllCall(SHWINLink;"WmemMoveMemory";:VOID;
        {Address(Pull);LPMemory+Offset;LoWord(DataSize)})
    RETURN(Pull)
ENDFUNC

//*****************************************************************
//*  PROCEDURE: GlobalFree
//*  DESC: Free a memory structure
//*****************************************************************
PROCEDURE GlobalFree(hMemory)
    DllCall(WINLink;"GlobalFree";Junk:Word;
        {LoWord(hMemory)})
```

```
ENDPROC

//*****************************************************************
//*  FUNCTION: LBGetCurSel
//*  DESC:
//*****************************************************************
FUNCTION LBGetCurSel(hWnd)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1033);LoWord(0);0})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: LBSetCurSel
//*  DESC: Set slection in a list box
//*****************************************************************
FUNCTION LBSetCurSel(hWnd;Index)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1031);LoWord(Index);0})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: LBGetSel
//*  DESC: Get index of selected item
//*****************************************************************
FUNCTION LBGetSel(hWnd;Index)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1032);LoWord(Index);0})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: LBSetSel
//*  DESC: Set index as selected item
//*****************************************************************
FUNCTION LBSetSel(hWnd;Select;Index)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1030);LoWord(Select);Index})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: LBGetText
//*  DESC:
//*****************************************************************
```

```
FUNCTION LBGetText(hWnd;Index)
    TextOut:=""
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1034);LoWord(Index);Address(TextOut)})
    RETURN(TextOut)
ENDFUNC

//*************************************************************
//*  FUNCTION: LBGetWPText
//*  DESC:
//*************************************************************
FUNCTION LBGetWPText(hWnd;Index)
    TextOut:=""
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1034);LoWord(Index);WPString(Address(TextOut))})
    RETURN(TextOut)
ENDFUNC

//*************************************************************
//*  FUNCTION: LBGetCount
//*  DESC:
//*************************************************************
FUNCTION LBGetCount(hWnd)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1036);LoWord(0);0})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  PROCEDURE: LBDeleteString
//*  DESC: Delete a text string from a list box.
//*************************************************************
PROCEDURE LBDeleteString(hWnd;Index)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1027);LoWord(Index);0})
ENDPROC

//*************************************************************
//*  PROCEDURE: LBResetContent
//*  DESC: Delete a text string from a list box.
//*************************************************************
PROCEDURE LBResetContent(hWnd)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1029);LoWord(0);0})
ENDPROC
```

```
//*************************************************************
//*  FUNCTION: LBAddString
//*  DESC: Add string to the combo box.
//*************************************************************
FUNCTION LBAddString(hWnd;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1025);LoWord(-1);Address(TextIn)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  PROCEDURE: LBInsertString
//*  DESC: Insert a text string into a list box.
//*************************************************************
PROCEDURE LBInsertString(hWnd;Index;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1026);LoWord(Index);Address(TextIn)})
ENDPROC

//*************************************************************
//*  PROCEDURE: LBInsertWPString
//*  DESC: Insert a text string into a WordPerfect list box.
//*************************************************************
PROCEDURE LBInsertWPString(hWnd;Index;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1026);LoWord(Index);WPString(Address(TextIn))})
ENDPROC

//*************************************************************
//*  FUNCTION: LBFindString
//*  DESC: Find a text string in a list box.
//*************************************************************
FUNCTION LBFindString(hWnd;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1040);LoWord(-1);Address(TextIn)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: CBAddString
//*  DESC: Add string to the combo box.
//*************************************************************
FUNCTION CBAddString(hWnd;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1027);LoWord(-1);Address(TextIn)})
    RETURN(Result)
```

```
ENDFUNC

//*****************************************************************
//*  PROCEDURE: CBInsertString
//*  DESC: Insert a text string into a combo box.
//*****************************************************************
PROCEDURE CBInsertString(hWnd;Index;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1034);LoWord(Index);Address(TextIn)})
ENDPROC

//*****************************************************************
//*  FUNCTION: CBGetLBText
//*  DESC:
//*****************************************************************
FUNCTION CBGetLBText(hWnd;Index)
    TextOut:=""
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1032);LoWord(Index);Address(TextOut)})
    RETURN(TextOut)
ENDFUNC

//*****************************************************************
//*  FUNCTION: CBGetCount
//*  DESC:
//*****************************************************************
FUNCTION CBGetCount(hWnd)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1030);LoWord(0);0})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: CBFindString
//*  DESC: Find a text string in a list box.
//*****************************************************************
FUNCTION CBFindString(hWnd;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1036);LoWord(-1);Address(TextIn)})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: LBGetSelCount
//*  DESC:
//*****************************************************************
```

```
FUNCTION LBGetSelCount(hWnd)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1041);LoWord(0);0})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  FUNCTION: LBGetSelItems
//*  DESC:
//*****************************************************************
FUNCTION LBGetSelItems(hWnd;NumItems;LPMemory)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(1042);LoWord(NumItems);LPMemory})
    RETURN(Result)
ENDFUNC

//*****************************************************************
//*  PROCEDURE: WMSetText
//*  DESC: Change text of a dialog control
//*****************************************************************
PROCEDURE WMSetText(hWnd;TextIn)
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWnd);LoWord(12);LoWord(0);Address(TextIn)})
ENDPROC

//*****************************************************************
//*  FUNCTION: WMGetText
//*  DESC: Get text of a dialog control
//*****************************************************************
FUNCTION WMGetText(hWnd)
    TextOut:=""
    DllCall(USERLink;"SendMessage";Result:DWord;
        {LoWord(hWnd);LoWord(13);LoWord(1024);
        Address(AnsiString(TextOut))})
    RETURN(TextOut)
ENDFUNC

//*****************************************************************
//*  PROCEDURE: EnableWindow
//*  DESC: Gray or un-gray a dialog control
//*****************************************************************
PROCEDURE EnableWindow(hWnd;State)
    DllCall(USERLink;"EnableWindow";Result:Bool;
        {LoWord(hWnd);LoWord(State)})
ENDPROC
```

```
//*************************************************************
//*  FUNCTION: GetFocus
//*  DESC: Get focus
//*************************************************************
FUNCTION GetFocus()
    DllCall(USERLink;"GetFocus";hWnd:Word;{})
    RETURN(hWnd)
ENDFUNC

//*************************************************************
//*  FUNCTION: SetFocus
//*  DESC: Get focus
//*************************************************************
FUNCTION SetFocus(hWnd)
    DllCall(USERLink;"SetFocus";hWnd:Word;{LoWord(hWnd)})
    RETURN(hWnd)
ENDFUNC

//*************************************************************
//*  FUNCTION: SetParent
//*  DESC: Change parent window of a child window
//*************************************************************
FUNCTION SetParent(hWndChild;hWndNewParent)
    DllCall(USERLink;"SetParent";Result:Word;
        {LoWord(hWndChild);LoWord(hWndNewParent)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: GetParent
//*  DESC: Get the parent window
//*************************************************************
FUNCTION GetParent(hWnd)
    While(hWnd <> 0)
        hWndParent:=hWnd
        DllCall(USERLink;"GetParent";hWnd:Word;
            {LoWord(hWndParent)})
    EndWhile
    RETURN(hWndParent)
ENDFUNC

//*************************************************************
//*  FUNCTION: IsIconic
//*  DESC: Check to see if a window is iconized
//*************************************************************
FUNCTION IsIconic(hWnd)
```

```
    DllCall(USERLink;"IsIconic";Result:Bool;
        {LoWord(hWnd)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: ShowWindow
//*  DESC: Change the way a window is shown
//*************************************************************
FUNCTION ShowWindow(hWnd;State)
    DllCall(USERLink;"ShowWindow";Result:Bool;
        {LoWord(hWnd);LoWord(State)})
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  PROCEDURE: ConvertToWP
//*  DESC: Change text of a dialog control
//*************************************************************
PROCEDURE ConvertToWP(TaskFile)
    CNVTLink:=0
    DllLoad(CNVTLink;LibraryPath+"shwinx20.dll")
        DllCall(CNVTLink;"WshxConvertFile";Result:WORD;
        {AnsiString(TaskFile);LoWord(1020);AnsiString("TempFile");LoWord(4);LoWord(0);0}
        )

    DllFree(CNVTLink)
    CopyFile("TempFile";TaskFile;NoPrompts!)
    DeleteFile("TempFile";NoPrompts!)
ENDPROC

//*************************************************
//   FUNCTION: InitializePIDFile
//   INPUTS: none
//   OUTPUTS: Result
//   USES: none
//   DESC:
//*************************************************
FUNCTION InitializePIDFile(AppID)
    Result:=0
    Group:="WP Macros"
    KeyPath:="WPConfig" SubKey:="" ExePath:=""
    Switch(AppID)
    CaseOf 2:
        ProdName:="WordPerfect"
        Section:="WordPerfect"
```

```
        PidFile:="wpwp61us.dll"
        SubKey1:="Apps\WPWin61\ExeLocation"        // WordPerfect
        SubKey2:="Apps\WPWin61\WorkStation\ExeLocation"
        ProdFile:="wpwin61.exe"
    CaseOf 3:
        ProdName:="Presentations"
        Section:="WPPrWin"
        PidFile:="wppr30us.dll"
        SubKey1:="Apps\PRWin30\ExeLocation"        // Presentations
        SubKey2:="Apps\PRWin30\WorkStation\ExeLocation"
        ProdFile:="prwin30.exe"
    CaseOf 4:
        ProdName:="GroupWise"
        Section:="WPOffice"
        PidFile:="wpofus.dll"
        SubKey1:="Apps\OFWin41\ExeLocation"        // Group Wise
        SubKey2:="Apps\OFWin41\WorkStation\ExeLocation"
        ProdFile:="ofwinfil.exe"
    CaseOf 5:
        ProdName:="Quattro Pro"
        Section:="QuattroPro"
        PidFile:="wpqp10us.dll"
        SubKey1:="Apps\QPW6.X\ExeLocation"         // Quattro Pro
        SubKey2:="Apps\QPW6.X\WorkStation\ExeLocation"
        ProdFile:="qpwin.exe"
    Default:
        RETURN(Result)
    EndSwitch
    DllCall(WPEXLink;"QueryReg";Result:INTEGER;
        {Address(KeyPath);Address(SubKey1);Address(ExePath)})
    If(ExePath = "")
        DllCall(WPEXLink;"QueryReg";Result:INTEGER;
            {Address(KeyPath);Address(SubKey2);Address(ExePath)})
    Endif
    If(NOT DoesFileExist(Filename:ExePath))            // check for stub .exe file
        If(ExePath = "")
            MessageBox(Caption:"Warning";
            Message:"A .EXE location could not be found for "+ToUpper(ProdName)+
            " in the registration database.";Style:OK!+IconInformation!)
        Else
            MessageBox(Caption:"File Not Found";
            Message:"File "+ToUpper(ExePath)+
            " not found.";Style:OK!+IconInformation!)
        Endif
        Result:=-1
        RETURN(Result)
```

```
Endif
OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
WfsPathSplit(ExePath;&OutDrive;&OutPathname;&OutFile;&OutExt)
ProgramPath:=OutDrive+OutPathname+ProdFile     // pathname plus filename
If(AppID = 5)                                  // if Quattro Pro
    ProgramPath:=WutlGetSharedPath()+ProdFile
Endif
If(NOT DoesFileExist(Filename:ProgramPath))    // check for main program .exe file
    MessageBox(Caption:"File Not Found";
    Message:"File "+ToUpper(ProgramPath)+
    " not found.";Style:OK!+IconInformation!)
    Result:=-1
    RETURN(Result)
Endif
If(CheckPidFile(Group;Section) = 2)
    Result:=2
    RETURN(Result)
Endif
FilePath1:=OutDrive+OutPathname+PidFile      // pathname plus filename
FilePath:=WutlGetSharedPath()
FilePath2:=FilePath+PidFile                  // shared code path plus filename
If(AppID = 5)                          // if Quattro Pro
    FilePath:=FilePath1
    FilePath1:=FilePath2               // check shared code path first for Quattro Pro
    FilePath2:=FilePath
Endif
PIDFound:=False
If(DoesFileExist(Filename:FilePath1))
    FilePath:=FilePath1
    PIDFound:=True
Else
    If(DoesFileExist(Filename:FilePath2))
        FilePath:=FilePath2
        PIDFound:=True
    Else
        MessageBox(Caption:"File Not Found";
        Message:"File "+ToUpper(FilePath1)+
        " not found.";Style:OK!+IconInformation!)
        Result:=-1
    Endif
Endif
If(PIDFound)
    BIFWrite(
        Status:StatusVar;
        Group:Group;
        Section:Section;
```

```
            Item:"UK";
            Value:ToUpper(FilePath);
            ItemType:AnsiString!;
            ItemFlags:1
        )
        BIFWrite(
            Status:StatusVar;
            Group:"WPMacroFacility";
            Section:Section;
            Item:"Command Line";
            Value:ToUpper(ProgramPath);
            ItemType:AnsiString!;
            ItemFlags:1
        )
        Result:=1
    Endif
    RETURN(Result)
ENDFUNC

//************************************************
//  FUNCTION: CheckPidFile
//  INPUTS: none
//  OUTPUTS: Group;Section
//  USES: none
//  DESC:
//************************************************
FUNCTION CheckPidFile(Group;Section)
    Result:=0
    BIFRead(
        Status:StatusVar;
        Group:Group;
        Section:Section;
        Item:"UK";
        Value:PidFile;
        ExpectedItemType:AnsiString!;
        NameListType:Private!
    )
    If(StatusVar > 0)
        Result:=1
        If(DoesFileExist(Filename:PidFile))
            Result:=2
        Endif
    Endif
    RETURN(Result)
ENDFUNC
```

```
//************************************************
//   FUNCTION: LoadWaitCursor
//   INPUTS: hWnd
//   OUTPUTS: Result
//   DESC:  Load hour glass cursor
//************************************************
FUNCTION LoadWaitCursor(hWnd)
    DllLoad(hLink;"USER")
    DllCall(hLink;"SetCapture";Result:Word;{LoWord(hWnd)})
    DllCall(hLink;"LoadCursor";Result:Word;{LoWord(0);32514})
    DllCall(hLink;"SetCursor";hCursor:Word;{LoWord(Result)})
    DllFree(hLink)
    RETURN(hCursor)
ENDFUNC

//************************************************
//   FUNCTION: FreeWaitCursor
//   INPUTS: hCursor
//   OUTPUTS: Result
//   DESC:  Free hour glass cursor and load original cursor
//************************************************
FUNCTION FreeWaitCursor(hCursor)
    DllLoad(hLink;"USER")
    DllCall(hLink;"SetCursor";Result:Word;{LoWord(hCursor)})
    DllCall(hLink;"ReleaseCapture";:VOID;{})
    DllFree(hLink)
    RETURN(Result)
ENDFUNC

//************************************************************
//*  FUNCTION: ChangeDefRect
//*  INPUTS: hWndDlg;hWndDef;hWndNew
//*  OUTPUTS: hWndDef
//*  DESC: Change the default button rectangle on a dialog
//************************************************************
FUNCTION ChangeDefRect(hWndDlg;hWndDef;hWndNew)
    If(hWndDef <> 0)
        // Get the default control ID that has the default rectangle
        DllCall(USERLink;"SendMessage";DefCtrlID:Integer;
            {LoWord(hWndDlg);LoWord(1024);LoWord(0);0})// DM_GETDEFID
        DefCtrlID:=GetLoWord(DefCtrlID)
        // Get the hWnd of the control
        DllCall(USERLink;"GetDlgItem";hWndDef:Word;
            {LoWord(hWndDlg);LoWord(DefCtrlID)})
    Endif
```

```
    // Get the control ID of the new control
    DllCall(USERLink;"GetDlgCtrlId";NewCtrlID:Word;
        {LoWord(hWndNew)})
    // Set the new dialog default rectangle
    DllCall(USERLink;"SendMessage";Result:Integer;
    {LoWord(hWndDlg);LoWord(1025);LoWord(NewCtrlID);0})  // DM_SETDEFID

    // Change the default control to that of a regular border
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWndDef);LoWord(1028);LoWord(0);1})// BM_SETSTYLE
    // Change the new default control to that of a default border
    DllCall(USERLink;"SendMessage";Result:Integer;
        {LoWord(hWndNew);LoWord(1028);LoWord(1);1})    // BM_SETSTYLE
    RETURN(hWndDef)
ENDFUNC

//*************************************************************
//*  FUNCTION: GetLoWord
//*  INPUTS: L
//*  DESC: Returns the lo order word of a 32 bit word
//*************************************************************
FUNCTION GetLoWord(L)
    RETURN (L & 0FFFFx)
ENDFUNC

//*************************************************************
//*  FUNCTION: GetHiWord
//*  INPUTS: L
//*  DESC: Return the hi order word of a 32 bit word
//*************************************************************
FUNCTION GetHiWord(L)
    RETURN (L >> 16)
ENDFUNC

//*************************************************************
//*  FUNCTION: DirectoryDlg
//*  OUTPUT: Result;&Pathname
//*  DESC: Directory select dialog
//*************************************************************
FUNCTION DirectoryDlg(&Pathname)
    DialogDefine("DirectoryDlg";
        50;50;200;80;
        19;"Directory")
    DialogAddText("DirectoryDlg";"DirText";
        8;8;182;12;
        0;"Current Directory:")
```

```
    DialogAddFilenameBox("DirectoryDlg";"Dirnm01";
        8;20;178;15;
        1;Pathname;Pathname;"*.*")
    DialogDisplay("DirectoryDlg";"Dirnm01")
    Result:=MacroDialogResult
    DialogDestroy("DirectoryDlg")
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: FileSpecDlg
//*  OUTPUT: Result;&Pathname
//*  DESC: Directory select dialog with static prompt text
//*************************************************************
FUNCTION FileSpecDlg(&Pathname;Title;StaticText;PromptText;Style)
    If(PromptText = "")
        Adjust:=0
        DialogDefine("FileSpecDlg";
            50;50;200;80;
            19;Title)
    Else
        Adjust:=17
        If(StrLen(PromptText) > 56)
            Adjust:=29
        Endif
        DialogDefine("FileSpecDlg";
            50;50;200;80+Adjust;
            19;Title)
        DialogAddText("FileSpecDlg";"PrmptText";
            8;8;182;24;
            0;PromptText)
    Endif
    DialogAddText("FileSpecDlg";"StaticText";
        8;8+Adjust;182;12;
        0;StaticText)
    DialogAddFilenameBox("FileSpecDlg";"Dirnm01";
        8;20+Adjust;178;15;
        Style;Pathname;Pathname;"*.*")
    DialogDisplay("FileSpecDlg";"Dirnm01")
    Result:=MacroDialogResult
    DialogDestroy("FileSpecDlg")
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: ShellExecute
```

```
//*  DESC: Execute a file
//*****************************************************************
FUNCTION ShellExecute(Filename)
    DllLoad(hLink;"SHELL")
    DllCall(hLink;"ShellExecute";Result:Word;
        {LoWord(0);0;Filename;0;0;LoWord(1)})

    If(Result < 32)
        Switch(Result)
        CaseOf 0:
            ErrMsg:="System was out of memory, executable file was corrupt, or relocations
            were invalid."
        CaseOf 2:
            ErrMsg:="File "+ToUpper(Filename)+" was not found."
        CaseOf 3:
            Filename:=Pathname(Filename)
            ErrMsg:="Path "+ToUpper(Filename)+" was not found."
        CaseOf 5:
            ErrMsg:="Attempt was made to dynamically link to a task, or there was a sharing or
            network-protection error."
        CaseOf 6:
            ErrMsg:="Library required separate data segments for each task."
        CaseOf 8:
            ErrMsg:="There was insufficient memory to start the application."
        CaseOf 10:
            ErrMsg:="Windows version was incorrect."
        CaseOf 11:
            ErrMsg:="Executable file was invalid. Either it was not a Windows application or
            there was an error in the .EXE image."
        CaseOf 12:
            ErrMsg:="Application was designed for a different operating system."
        CaseOf 13:
            ErrMsg:="Application was designed for MS-DOS 4.0."
        CaseOf 14:
            ErrMsg:="Type of executable file was unknown."
        CaseOf 15:
            ErrMsg:="Attempt was made to load a real-mode application (developed for an
            earlier version of Windows)."
        CaseOf 16:
            ErrMsg:="Attempt was made to load a second instance of an executable file
            containing multiple data segments that were not marked read-only."
        CaseOf 19:
            ErrMsg:="Attempt was made to load a compressed executable file. The file must be
            decompressed before it can be loaded."
        CaseOf 20:
            ErrMsg:="Dynamic-link library (DLL) file was invalid. One of the DLLs required to
```

```
            run this application was corrupt."
        CaseOf 21:
            ErrMsg:="Application requires Microsoft Windows 32-bit extensions."
        Default:
            ErrMsg:=""
            Result:=31

        EndSwitch
        If(ErrMsg <> "")
            MessageBox(Caption:"File Error";
            Message:ErrMsg;Style:OK!+IconInformation!)
        Endif
    Endif
    DllFree(hLink)
    RETURN(Result)
ENDFUNC

//*************************************************************
//*  FUNCTION: ValidateDirectory
//*  DESC: Open a file to check read and write rights
//*  OUTPUT: ValidDirectory
//*  USES: CheckBackslash;DirectoryDlg
//*************************************************************
FUNCTION ValidateDirectory(&Pathname)
    ValidDirectory:=False
    // Check to see if the directory provided is valid
    // (ie.  If this directory is located on a network the user may only have read/scan)
    While(NOT ValidDirectory)
        Pathname:=CheckBackslash(Pathname)
        // Try to open a file in the current directory
        If(ValidFileIO(Pathname))
            ValidDirectory:=True
            BREAK
        Else
            // NO!  not successful in opening the file
            HRt = NTOC(0F90Ah)
            If(DoesDirectoryExist(DirectoryName:Pathname))
                // Prompt the user if they want to select another directory
                MessageBox(Status:Result;Caption:"Invalid Rights";
                Message:"Directory  "+ToUpper(Pathname)+" cannot be written to.  Do you wish
                to "+HRt+"select another directory?";Style:YesNo!+IconQuestion!)
            Else
                MessageBox(Status:Result;Caption:"Invalid Directory";
                Message:"Directory  "+ToUpper(Pathname)+" does not exist.  Do you wish to
                "+HRt+"select another directory?";Style:YesNo!+IconQuestion!)
            Endif
```

```
            If(Result = 6)
                // YES! the user DOES want to select  another directory
                // Bring up the directory select dialog
                Result = DirectoryDlg(&Pathname)
                If(Result = 2)
                    BREAK
                Endif
            Else
                BREAK
            Endif
        Endif
    EndWhile
    RETURN(ValidDirectory)
ENDFUNC

//*************************************************************
//*  FUNCTION: SelectDirectory
//*  INPUTS: Pathname
//*  OUTPUTS: ValidDir
//*  DESC: Creates and validates a directory
//*************************************************************
FUNCTION SelectDirectory(&Pathname;Title;PromptText)
    ValidDir:=True
    DirectoryExists:=False
    If(DoesDirectoryExist(DirectoryName:Pathname))
        DirectoryExists:=True
    Endif
    ValidDirectory:=False
    CreateFailed:=False
    If(ValidFileIO(Pathname))
        ValidDirectory:=True
    Endif
    While(NOT DirectoryExists or NOT ValidDirectory)
        If(DirectoryExists and NOT ValidDirectory and NOT CreateFailed)
            HRt = NTOC(0F90Ah)
            MessageBox(Status:Result;Caption:"Invalid Rights";
            Message:"Directory  "+ToUpper(Pathname)+" cannot be written to.  Do you wish to
            "+HRt+"select another directory?";Style:YesNo!+IconQuestion!)
            If(Result = 7)
                ValidDir:=False
                RETURN(ValidDir)
            Endif
        Endif
        If(PromptText <> "")
            Result:=FileSpecDlg(&Pathname;Title;"Create Directory:";PromptText;1)
        Else
```

```
            Result:=FileSpecDlg(&Pathname;Title;"Create Directory:";"";1)
        Endif
        If(Result = 1)
            If(Pathname <> "")
                OutDrive:="" OutPathname:="" OutFile:="" OutExt:=""
                WfsPathSplit(Pathname;&OutDrive;&OutPathname;&OutFile;&OutExt)
                If(OutDrive = "" and OutPathname = "")
                    FilePath:=""
                    DllCall(SHWINLink;"WfsGetCurDir";Result:INTEGER;
                        {Address(FilePath)})
                    FilePath:=CheckBackslash(FilePath)
                    Pathname:=FilePath+OutFile+OutExt
                Endif
                CreateDirectory(DirectoryName:Pathname;Prompts:NoPrompts!)
                CreateFailed:=False
                DirectoryExists:=False
                If(DoesDirectoryExist(DirectoryName:Pathname))
                    DirectoryExists:=True
                Else
                    MessageBox(Caption:"Invalid Rights";
                    Message:"Directory "+ToUpper(Pathname)+" could not be
                    created.";Style:OK!+IconInformation!)
                    CreateFailed:=True
                Endif
                If(ValidFileIO(Pathname))
                    ValidDirectory:=True
                Endif
            Endif
        Else
            ValidDir:=False
            BREAK
        Endif
    EndWhile
    RETURN(ValidDir)
ENDFUNC

//*************************************************************
//*  FUNCTION: ValidFileIO
//*  INPUTS: none
//*  OUTPUTS: Result
//*************************************************************
FUNCTION ValidFileIO(Pathname)
    Result:=False
    hFile:=OpenFile(Name:Pathname+"FILE_IO.DOC";AccessMode:Write!)
    If(hFile > 0)
        // Yes! successful in opening the file
```

```
        // Close the file opened then delete the file and continue
        CLOSEFILE(FileID:hFile)
        DELETEFILE(Filename:Pathname+"FILE_IO.DOC")
        Result:=True
    Endif
    RETURN(Result)
ENDFUNC
```