```
//************************************************************
//   MACRO: EXPSHOWS.WCM
//   DESCRIPTION: Show Expert macro, help the user create a presentation,
using predefined outlines.
//************************************************************
Application (A1; "WPPrWin"; Default; "US")
//************************************************************
//   MAIN PROGRAM
//************************************************************
Call(Init@)    // Modify this label to customize.
Call(DlgWelcome@)
Call(DlgTab@)
Call(FreeBMP@)
Call(AssignTopics@)
Call(InsertOutline@)
Call(DlgHelp@)
Go(Quit@)
//************************************************************
//   MAIN PROGRAM END
//************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: Init@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(Init@)
AppName := "Show Expert"
Call(WaitMsgDisplay@)
DialogHandle(hWait; "Wait1")
DialogHandle(hWaitText; "Wait1";"WaitText")

hPRWin := AppLocate("Presentations 3.*")

DLLLoad(Ulink;"USER")
DLLLoad(GLink;"GDI")

hCursor := LoadWaitCursor(ULink; hPRWin)

//-------------------------------------------------------------
// Directory for Shows Expert files (DLL, HLP, Outlines)
//-------------------------------------------------------------
OutlineDir := EnvPaths(13) //ShowExpert!

//-------------------------------------------------------------
// Array variable to contain information for presentations.
// Increase the last size for more presentation items.
//   Default:  Declare TCon[4,15]
//   To:     Declare TCon[4,31]  // for 10 items
// This number should be (MaxNumberOfItems * 3) + 1
//-------------------------------------------------------------
Declare TCon[4,15]     // Array for Tab content
//-------------------------------------------------------------
// Define TCon[]
// Format for first parameter:
//  TCon[1,x] = Inform presentation type
//  TCon[2,x] = Persuade presentation type
//  TCon[3,x] = Teach presentation type
//  TCon[4,x] = Special occasion presentation type
//
// Format for second parameter:
//  TCon[x,1] = Number of items for this presentation type
//
//  TCon[x,2] = First Menu description
//  TCon[x,3] = First outline filename
//  TCon[x,4] = First Description/help text
//
//  TCon[x,5] = Second Menu description
//  TCon[x,6] = Second outline filename
//  TCon[x,7] = Second Description/help text
//  Continue this format up to the number of items.
```

```
//---------------------------------------------------------------
// Inform presentation type Information
//---------------------------------------------------------------
TCon[1,1] := 3    // number of items for inform
// Inform Item 1
TCon[1,2] := "Report Results"
TCon[1,3] := "in_resul.wpd"
TCon[1,4] :=   "This presentation tells listeners about results achieved or
progress made."
// Inform Item 2
TCon[1,5] :=   "Present idea(s)"
TCon[1,6] := "in_ideas.wpd"
TCon[1,7] :=   "This presentation tells listeners about an idea that may be
new to them.  It explains the idea as it may apply to them."
// Inform Item 3
TCon[1,8] :=   "Describe alternatives"
TCon[1,9] := "in_alter.wpd"
TCon[1,10]:=   "This presentation tells listeners about two or more options
they can choose from."

//---------------------------------------------------------------
// Persuade presentation type information
//---------------------------------------------------------------
TCon[2,1] := 4    // number of items for persuade
// Persuade item 1
TCon[2,2] := "Positive Audience"
TCon[2,3] := "pe_posi.wpd"
TCon[2,4] := "This presentation prepares the listener to purchase a product
or service, or to understand why a change is necessary.  Use this
presentation for an accepting or positive audience."
// Persuade item 2
TCon[2,5] := "Negative Audience"
TCon[2,6] := "pe_nega.wpd"
TCon[2,7] := "This presentation prepares the listener to purchase a product
or service, or to understand why a change is necessary.  Use this
presentation for a resisting or negative audience."
// Persuade item 3
TCon[2,8] := "Doubting Audience"
TCon[2,9] := "pe_doub.wpd"
TCon[2,10]:= "This presentation prepares the listener to purchase a product
or service, or to understand why a change is necessary.  Use this
presentation for a skeptical or doubting audience."
// Persuade item 4
TCon[2,11]:= "Neutral Audience"
TCon[2,12]:= "pe_neut.wpd"
TCon[2,13]:= "This presentation prepares the listener to purchase a product
or service, or to understand why a change is necessary.  Use this
presentation for an indifferent or neutral audience."

//---------------------------------------------------------------
// Teach presentation type
//---------------------------------------------------------------
```

```
TCon[3,1] := 2    // number of items for tech
// Teach item 1
TCon[3,2] := "Teach a skill"
TCon[3,3] := "te_teach.wpd"
TCon[3,4] := "This presentation teaches skill development.  It prepares the
participant to learn the skill, sets skill patterns in the participant's
mind, helps the participant to perform and practice the skill, and reviews
the learner's performance."
// Teach item 1
TCon[3,5] := "Explain a concept or method"
TCon[3,6] := "te_expln.wpd"
TCon[3,7] := "This presentation explains a concept, idea, or approach."

//---------------------------------------------------------------
// Special Occasion presentation type
//---------------------------------------------------------------
TCon[4,1] := 2    // number of items for special occasion
// Special Occation item 1
TCon[4,2] := "Present an award or tribute"
TCon[4,3] := "sp_award.wpd"
TCon[4,4] := "This presentation is used when you give someone an award."
// Special Occation item 2
TCon[4,5]:=  "Welcome or introduce"
TCon[4,6]:=  "sp_welco.wpd"
TCon[4,7]:=  "This presentation welcomes an audience to an event,
introduces a presenter, or both."

// Init other variables
Declare TRV[4,4]    // Array for Tab Return Variables
Declare TDlg[4,10]  // Array for Tab Handles
Declare THlp[4]     // Array for Help values
THlp[] := {261;262;263;264}
TDlg[1,2] := "Inform"
TDlg[2,2] := "Persuade"
TDlg[3,2] := "Teach"
TDlg[4,2] := "Special"
InitDlg   := 1
OKBttn    := 0
OldIndex  := 0

WelDlg := 1
BIFRead(stat;"WPPrWin"; "Show Expert"; "Display Welcome";
        WelDlg; SignedDWord!;"|";Private!)

HlpDlg := 1
BIFRead(stat;"WPPrWin"; "Show Expert"; "Display Additional Help";
        HlpDlg; SignedDWord!;"|";Private!)


dn:=100
HelpFile :=EnvPaths(Program!) + "WPPRUS.HLP"
DLLLoad(Link;OutlineDir + "expshows.dll")
```

```
hFont   := SelectFont("Arial"; 14)
hFont16 := SelectBoldFont("Arial"; 16)
hFont18 := SelectBoldFont("Arial"; 18)
hFont24 := SelectFont("Arial"; 24)

Call(LoadBMP@)

RETURN
//*************************************************************
```

```
//*****************************************************************
//   ROUTINE NAME: AssignTopics@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION: Assign Topics to Array.
//*****************************************************************
LABEL(AssignTopics@)
// Task[ x, 5] = 1:Topic, 2:Description,
//   3:SlideType, 4:NumOfSlides
PresType := TCon[CurTab,c]

Switch (CurTab)
     CaseOf 1 : // Inform
        Switch (PresType)
           CaseOf "Report Results":
              H_ITEM := 265
           CaseOf "Present idea(s)":
              H_ITEM := 266
           CaseOf "Describe alternatives":
              H_ITEM := 267
        EndSwitch
     CaseOf 2 : // Pursuade
        Switch (PresType)
           CaseOf "Positive Audience":
              H_ITEM := 268
           CaseOf "Negative Audience":
              H_ITEM := 269
           CaseOf "Doubting Audience":
              H_ITEM := 270
           CaseOf "Neutral Audience":
              H_ITEM := 271
        EndSwitch
     CaseOf 3 : // Teach
        Switch (PresType)
           CaseOf "Teach a skill":
              H_ITEM := 272
           CaseOf "Explain a concept or method":
              H_ITEM := 273
        EndSwitch
     CaseOf 4 : // Special
        Switch (PresType)
           CaseOf "Entertain":
              H_ITEM := 274
           CaseOf "Present and award":
              H_ITEM := 275
           CaseOf "Present a tribute":
              H_ITEM := 276
           CaseOf "Welcome or introduce":
              H_ITEM := 277
        EndSwitch
EndSwitch
```

```
RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: Quit@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(Quit@)

BIFWrite(stat;"WPPrWin"; "Show Expert"; "Display Welcome";
       WelDlg; SignedDWord!;None!)
BIFWrite(stat;"WPPrWin"; "Show Expert"; "Display Additional Help";
       HlpDlg; SignedDWord!;None!)

DLLCall(ULink;"SetFocus";nu;VOID;{LOWORD(hPRWin)})

FreeFont(hFont)
FreeFont(hFont16)
FreeFont(hFont18)
FreeFont(hFont24)

DLLFree(link)
DLLFree(ULink)
DLLFree(GLink)

QUIT
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: FreeBMP@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(FreeBMP@)

DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hTabBMP)})
For ( x; 1; x < 5; x + 1)
     DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(TDlg[x,1])})
EndFor

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: DlgWelcome@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(DlgWelcome@)
if (WelDlg <> 1) RETURN endif

hrt := NTOC(10)
Wtext := "Welcome to Show Expert" + hrt + hrt + "Show Expert helps you
     create presentations.  All you need to do is select the type of
     presentation you want to give, and Show Expert helps you determine the
     best way to present your information." + hrt + hrt + "After you select
     a type of presentation, Show Expert provides an outline suggesting an
     effective way for you to present your information.  You just need to
     replace the sample text with your own text."
wdn := 200
DialogDefine(wdn;50;50;275;205;16;"Welcome to "+APPName)
DialogAddFrame(wdn;"frm"; 4;4;135;164;0)
DialogAddControl(wdn;"bmp";6;6;131;160;"WPbmp20"; 4; ""; test; 0)
DialogAddText(wdn;"t1";144;8;120;170;1;WText)
DialogAddPushButton(wdn;1;270/2 - 35/2;172;35;13;1;"OK")

DialogHandle(hwbmp;wdn;"bmp")

DllCall(ULink;"LoadBitmap";wbmp:WORD;{
     LOWORD(Link);  //hinstance
     "WBITMAP"      //bitmap id
})

DllCall(ULink;"SendMessage";ret:DWORD;{
     LOWORD(hwbmp);    //hwnd of the control
     LOWORD(1024+900); //BMPM_SETHBITMAP message
     LOWORD(wbmp);  //handle to bitmap
     0         //instance
})

// Load Normal Cursor
FreeWaitCursor(ULink; hCursor)

DialogDisplay(wdn;1)
DialogDestroy(wdn)

// Load Wait Cursor
hCursor := LoadWaitCursor(ULink; hPRWin)

DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(wbmp)})

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: DlgTab@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(DlgTab@)

DialogDefine(dn;50;50;178;175;16;APPName)

DialogAddText   (dn;"t0";   8;4;155;10;1;"Choose presentation style you
want.")

DialogAddHotSpot(dn;"h1";  1;1+16;42;15;1)
DialogAddHotSpot(dn;"h2"; 44;1+16;42;15;1)
DialogAddHotSpot(dn;"h3"; 88;1+16;42;15;1)
DialogAddHotSpot(dn;"h4";131;1+16;42;15;1)

DialogAddControl(dn;"bmp1";0;1+16;175;15;"WPbmp20"; 2; ""; test; 0)
DialogAddControl(dn;"bmp"; 0;1+16;175;15;"WPbmp20"; 2+8; ""; test; 0)

DialogAddText   (dn;"t1";  1+3;4+16;35;10;4;"")//9:36x10
DialogAddText   (dn;"t2"; 44+3;4+16;35;10;4;"")//9:36x10
DialogAddText   (dn;"t3"; 88+3;4+16;35;10;4;"")//9:36x10
DialogAddText   (dn;"t4";131+3;4+16;35;10;4;"")//6:36x15

DialogAddPushButton (dn;      1;132;25+16;35;13;1;"OK")
DialogAddPushButton (dn;      2;132;41+16;35;13;0;"Cancel")
DialogAddPushButton (dn;"Setup";132;60+16;35;13;0;"&Setup...")
DialogAddPushButton (dn;"Help" ;132;76+16;35;13;0;"&Help")
//--
Call(DlgTabCBInit@)
//--
FreeWaitCursor(ULink; hCursor)   // ????

DialogDisplay(dn;1;DlgTabCB@)

DoDlg := 1
While(DoDlg)
EndWhile
Button := MacroDialogResult
DialogDestroy(dn)

if ( Button = 1)
     OutlineName := TCon[CurTab,c+1]
endif

if ( Button = 2)
     GO(Quit@)
endif
```

```
ShowWin(ULink; hWait; 1)

hCursor := LoadWaitCursor(ULink; hPRWin)

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: DlgTabCBInit@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(DlgTabCBInit@)
     DialogHandle(hbmp1;dn;"bmp1")
     DialogHandle(hbmp;dn;"bmp")
     DialogHandle(TDlg[1;3];dn;"t1")
     DialogHandle(TDlg[2;3];dn;"t2")
     DialogHandle(TDlg[3;3];dn;"t3")
     DialogHandle(TDlg[4;3];dn;"t4")

     // Set Clear Color
     R:=255 G:=0 B:=255
     ColorRef := (R) | (G << 8) | (B << 16)

     DllCall(ULink;"SendMessage";ret:DWORD;{
        LOWORD(hbmp);        //hwnd of the control
        LOWORD(1024+902); //BMPM_SETTANSPARENTCOLOR message
        LOWORD(0);           //handle to bitmap
        ColorRef             //instance
     })
     //DllCall(ULink;"SendMessage";ret:DWORD;{
     // LOWORD(hbmp1);       //hwnd of the control
     // LOWORD(1024+902); //BMPM_SETTANSPARENTCOLOR message
     // LOWORD(0);           //handle to bitmap
     // ColorRef             //instance
     //})

     DllCall(ULink;"SendMessage";ret:DWORD;{
        LOWORD(hbmp);        //hwnd of the control
        LOWORD(1024+900); //BMPM_SETHBITMAP message
        LOWORD(TDlg[1;1]);   //handle to bitmap
        0                    //instance
     })

     DllCall(ULink;"SendMessage";ret:DWORD;{
        LOWORD(hbmp1);       //hwnd of the control
        LOWORD(1024+900); //BMPM_SETHBITMAP message
        LOWORD(hTabBMP);     //handle to bitmap
        0                    //instance
     })

     Call(SetTabText@)

     DialogAddText    (dn;"t_1";10;25+16;100;10;1;"Presentation Outline:")
     // DialogAddCheckBox(dn;"chk";10;140+16;142;10;"Display additional
help  in outline.";HlpDlg)
     DialogAddText    (dn;"dt0";10;83+16;70;10;1;"Description:")
```

```
    DialogAddHLine  (dn;"hl1"; 5;91+16;163)

    // Add listboxs and desc text
    for (CurTab;1;CurTab<5;CurTab+1)
       DialogAddListbox(dn;"l"+CurTab;10;36+16;112;40;1+2;TRV[CurTab,1])
       for ( c; 2; c <= TCon[CurTab,1] * 3; c+3)
          DialogAddListItem(dn;"l"+CurTab; TCon[CurTab,c])
       EndFor
       DialogAddText   (dn;"tx"+CurTab;10;93+16;153;45;1;"")

       DialogHandle(TDlg[CurTab,4];dn;"l"+CurTab)
       DialogHandle(TDlg[CurTab,5];dn;"tx"+CurTab)
       SetCtrlFont(dn;"tx"+CurTab; hFont)

       DLLCall(ULink;"SendMessage";ret:DWORD;{
          LOWORD(TDlg[CurTab,4]);  // hLBox
          LOWORD(1024+7);          // LB_SETCURSEL
          LOWORD(0);               // First Item
          0
       })
       OldIndex := 20
       CALL(SetDescText@)
       ShowWin ( Ulink; Tdlg[CurTab, 4]; 0)
       ShowWin ( Ulink; Tdlg[CurTab, 5]; 0)
    EndFor

RETURN
//*************************************************************
```

```
//*****************************************************************
//   ROUTINE NAME: DlgTabCB@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*****************************************************************
LABEL(DlgTabCB@)
If ( InitDlg)
     InitDlg := 0
     hwnd := DlgTabCB[4]
     ShowWin(ULink; hWait; 0)
     SetWinText(ULink; hWaitText; "Creating Presentation."+NTOC(10)+"Please
       Wait...")
     CurTab := 1
     Call(ShowTab@)
     SetFocus ( ULink; TDlg[CurTab,4])
EndIf

// HotSpot - Tab
If ( SubStr(DlgTabCB[3];1;1) = "h")
     x := SubStr(DlgTabCB[3];2;1)
     If (CurTab = x) RETURN EndIf
     DisplayTab( ULink; hbmp; TDlg[x,1])
     DisplayText( ULink; TDlg[x,3])
     Call(HideTab@)
     CurTab := x
     Call(ShowTab@)
     SetFocus ( ULink; TDlg[CurTab,4])
EndIf

// ListBox
If (SubStr(DlgTabCB[3];1;1) = "l")
     Call(SetDescText@)
     If ( DlgTabCB[8] = 2)
       OKBttn := 1
       FreeHelp(hwnd; HelpFile) //Call(FreeHelp@)
       DoDlg:=0
       DialogUndisplay(dn;1)
     EndIf
EndIf

// if SYS_CLOSE
if(DlgTabCB[5] = 274)
     FreeHelp(hwnd; HelpFile) //Call(FreeHelp@)
     DoDlg:=0
     DialogUndisplay(dn;2)
endif

Switch (DlgTabCB[3])
     CaseOf "Help" :
       LoadHelp(hwnd; HelpFile; THlp[CurTab]) //Call(LoadHelp@)
```

```
      CaseOf "Setup" :
         DlgSetup( &WelDlg; &HlpDlg)

      CaseOf 1 :
         OKBttn := 1
         Call(SetDescText@)
         FreeHelp(hwnd; HelpFile) //Call(FreeHelp@)
         DoDlg:=0
         DialogUndisplay(dn;1)

      CaseOf 2 :
         FreeHelp(hwnd; HelpFile) //Call(FreeHelp@)
         DoDlg:=0
         DialogUndisplay(dn;2)

EndSwitch

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: LoadBMP@
//   INPUT VARIABLES:
//   RETURN VARIABLES: hB1, hB2, hB3, hB4
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(LoadBMP@)

// CHECK FONT SIZE
DllCall(ULink;"GetSystemMetrics";fsize:WORD;{
     LOWORD(4) //bitmap id
})

if(fsize > 22)
     // Load Bitmaps for Tab Dlg
     DllCall(ULink;"LoadBitmap";hTabBMP:WORD;{
        LOWORD(Link);  //hinstance
        "BITMAP_L0"    //bitmap id
     })
     for (x;1;x<5;x+1)
        DllCall(ULink;"LoadBitmap";TDlg[x;1]:WORD;{
           LOWORD(Link);  //hinstance
           "BITMAP_L"+x //bitmap id
        })
     endfor
else
     // Load Bitmaps for Tab Dlg
     DllCall(ULink;"LoadBitmap";hTabBMP:WORD;{
        LOWORD(Link);  //hinstance
        "BITMAP_0"     //bitmap id
     })
     for (x;1;x<5;x+1)
        DllCall(ULink;"LoadBitmap";TDlg[x;1]:WORD;{
           LOWORD(Link);  //hinstance
           "BITMAP_"+x  //bitmap id
        })
     endfor
endif

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: HideTab@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(HideTab@)
// Hide Listbox
ShowWin( ULink; TDlg[CurTab,4]; 0)
// Hide Desc Text
ShowWin( ULink; TDlg[CurTab,5]; 0)

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: ShowTab@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(ShowTab@)
// Show ListBox
ShowWin ( ULink; TDlg[CurTab,4]; 1)
// Show Desc Text
ShowWin ( ULink; TDlg[CurTab,5]; 1)

OldIndex := 0
DLLCall(ULink;"SendMessage";OldIndex:DWORD;{
     LOWORD(TDlg[CurTab,4]);  // hLBox
     LOWORD(1024+9);          // LB_GETCURSEL
     LOWORD(0);               // NA
     0
})

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: SetTabText
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
Label(SetTabText@)

for (x;1;x<5;x+1)
     SetText ( ULink; TDlg[x,3]; TDlg[x,2])
EndFor

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: SetDescText@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(SetDescText@)
Index := 0
ItemText := "                                                    "
DLLCall(ULink;"SendMessage";Index:DWORD;{
     LOWORD(TDlg[CurTab,4]);  // hLBox
     LOWORD(1024+9);          // LB_GETCURSEL
     LOWORD(0);               // NA
     0
})
DLLCall(ULink;"SendMessage";ret:DWORD;{
     LOWORD(TDlg[CurTab,4]);  // hLBox
     LOWORD(1024+10);         // LB_GETTEXT
     LOWORD(Index);           // Item Index
     ADDRESS(ItemText)      // Variable for item
})

For ( c; 2; c <= TCon[CurTab, 1] * 3; c+3)
     If ( TCon[CurTab, c] = ItemText)
        If ( OKBttn = 0)
           If ( OldIndex <> Index)
              OldIndex := Index
              SetText ( ULink; TDlg[CurTab,5]; TCon[CurTab, c+2])
           EndIf
           ChangeTab := 0
        EndIf
        Break
     EndIf
EndFor

RETURN
//*************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: InsertOutline@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(InsertOutline@)
Discard TDlg[]
Discard TCon[]

SlideShowOutline ()
FileRetrieve ( OutlineDir + OutlineName; YES!)
BegOfLine ()

// change slide types
NumSlides := EnvNumberOfSlides

CurSlide := 0

ONERROR CALL(SlideTemplateError@)
ONNOTFOUND (SLideTemplateNotFound@)
ONCANCEL CALL(SLideTemplateError@)

while(CurSlide < NumSlides)

     TextSearch (SearchString: "]"; SearchDirection: Forward!;
               SearchBeep: TRUE)
     TextRight
     SelectBegOfLine
     SelText := EnvSelectedText

     CurSlide := CurSlide+1

     Switch(SelText)
       CaseOf "[Title] ":
          Delete
          SetTemplateRange(CurSlide; CurSlide; "Title")

       CaseOf "[Text] ":
          Delete
          SetTemplateRange(CurSlide; CurSlide; "Text")

       CaseOf "[Bullet] ":
          Delete
          SetTemplateRange(CurSlide; CurSlide; "Bullet Chart")

       CaseOf "[Data] ":
          Delete
          SetTemplateRange(CurSlide; CurSlide; "Data Chart")

       CaseOf "[Combo] ":
```

```
            Delete
            SetTemplateRange(CurSlide; CurSlide; "Combination")

         CaseOf "[Org] ":
            Delete
            SetTemplateRange(CurSlide; CurSlide;"Org Chart")

         Default:
            Delete

      EndSwitch

endwhile

LABEL(SLideTemplateNotFound@)

ONERROR()
ONNOTFOUND()
ONCANCEL()

DocumentTop ()

RETURN
//*****************************************************************
```

```
//*****************************************************
//   PROCEDURE: DlgSetup (&WDlg; &HDlg)
//   DESCRIPTION: Setup dlg
//*****************************************************
PROCEDURE DlgSetup (&WDlg; &HDlg)
Wtemp := Wdlg
Htemp := HDlg
ds:="DlgSetup"
DialogDefine(ds;50;50;186;59;16;"Show Expert Setup")
DialogAddCheckBox(ds;"ck1";8; 8;120;12;"Display welcome screen"; Wtemp)
DialogAddCheckBox(ds;"ck2";8;20;120;12;"Display additional help screen";
Htemp)

DialogAddPushButton(ds;1;140; 8;35;13;1;"OK")
DialogAddPushButton(ds;2;140;24;35;13;0;"Cancel")

DialogDisplay(ds;1)

// update global vars if OK
if ( MacroDialogResult = 1)
     Wdlg := Wtemp
     Hdlg := Htemp
endif

DialogDestroy(ds)
ENDPROC
//*****************************************************
```

```
//*************************************************************
//   ROUTINE NAME: DlgHelp@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(DlgHelp@)

Call(WaitMsgDestroy@)
if ( HlpDlg = 0) // Display Additional Help not checked
     FreeWaitCursor(ULink; hCursor)
     RETURN
EndIf

hCursor := LoadWaitCursor(ULink; hPRWin)

dn:=100
InitDlg := 1
DialogDefine (dn;100;50;150;245;8+16+32;AppName)

DialogAddText       (dn;  "t1"; 6;  4;127;10;1;"")
DialogAddText       (dn;  "t2"; 6; 16;137;35;1;"")
DialogAddText       (dn;  "t3"; 6; 56;127;10;1;"")
DialogAddText       (dn;  "t4"; 6; 67;137;35;1;"")
DialogAddText       (dn;  "t5"; 6;107;127;10;1;"")
DialogAddText       (dn;  "t6"; 6;118;137;45;1;"")
DialogAddText       (dn;  "t7"; 6;160;127;10;1;"")
DialogAddText       (dn;  "t8";30;175;100;10;1;"")
DialogAddText       (dn;  "t9";30;188;100;10;1;"")
DialogAddText       (dn; "t10";30;201;100;10;1;"")
DialogAddText       (dn; "t11";30;214;100;10;1;"")

DialogAddHotSpot(dn; "h1";11;172;14;14;1)
DialogAddHotSpot(dn; "h2";11;185;14;14;1)
DialogAddHotSpot(dn; "h3";11;199;14;14;1)
DialogAddHotSpot(dn; "h4";11;211;14;14;1)
//DialogAddControl(dn;"bmp";10;171;130;40;"WPbmp20"; 2; ""; test; 0)
DialogAddControl(dn;"bmp1";11;172;14;14;"WPbmp20"; 4; ""; test; 0)
DialogAddControl(dn;"bmp2";11;185;14;14;"WPbmp20"; 4; ""; test; 0)
DialogAddControl(dn;"bmp3";11;199;14;14;"WPbmp20"; 4; ""; test; 0)
DialogAddControl(dn;"bmp4";11;211;14;14;"WPbmp20"; 4; ""; test; 0)

DialogAddPushButton (dn;     1;110;195; 35;13;1;"&Close")
DialogAddPushButton (dn;"Help";110;211; 35;13;0;"&Help")

     //DialogHandle(hbmp;dn;"bmp")
     DialogHandle(hbmp1;dn;"bmp1")
     DialogHandle(hbmp2;dn;"bmp2")
     DialogHandle(hbmp3;dn;"bmp3")
     DialogHandle(hbmp4;dn;"bmp4")
     DialogHandle(ht1;dn;"t1")
```

```
      DialogHandle(ht2;dn;"t2")
      DialogHandle(ht3;dn;"t3")
      DialogHandle(ht4;dn;"t4")
      DialogHandle(ht5;dn;"t5")
      DialogHandle(ht6;dn;"t6")
      DialogHandle(ht7;dn;"t7")
      DialogHandle(ht8;dn;"t8")
      DialogHandle(ht9;dn;"t9")
      DialogHandle(ht10;dn;"t10")
      DialogHandle(ht11;dn;"t11")

      SetCtrlFont(dn;"t1";hFont18)
      SetCtrlFont(dn;"t2";hFont)
      SetCtrlFont(dn;"t3";hFont16)
      SetCtrlFont(dn;"t4";hFont)
      SetCtrlFont(dn;"t5";hFont16)
      SetCtrlFont(dn;"t6";hFont)
      SetCtrlFont(dn;"t7";hFont16)
      SetCtrlFont(dn;"t8";hFont)
      SetCtrlFont(dn;"t9";hFont)
      SetCtrlFont(dn;"t10";hFont)
      SetCtrlFont(dn;"t11";hFont)

      HRT := NTOC(10)
      SetText( ULink; ht1; "Additional Help:")
      SetText( ULink; ht2; "This is your presentation outline."+HRT+"The
        presentation can be edited from this outline view or from another
        view.  To edit the outline, select the text and replace it with
        your own.")
      SetText( ULink; ht3; "Outliner Key strokes:")
      SetText( ULink; ht4;
        "Tab           - Promotes the current line"+HRT+
        "Shift+Tab  - Demotes the current line"+HRT+
        "Enter         - Inserts a new line"+HRT+
        "Ctrl+Enter  - Inserts a new slide")
      SetText( ULink; ht5; "Changing views:")
      SetText( ULink; ht6; "To change views, select Slide Editor, Outliner,
        Slide List, or Slide Sorter from the View menu; or select one of
        the following view buttons from the Tool Palette at the left of
        your screen:")
      SetText(ULink; ht7;"View Buttons:")
      SetText(ULink; ht8;"Slide Editor")
      SetText(ULink; ht9;"Outliner")
      SetText(ULink;ht10;"Slide List")
      SetText(ULink;ht11;"Slide Sorter")

//DLLCall(ULink; "LoadBitmap"; hhlpbmp;WORD;{
//   LOWORD(Link);  //hinstance
//   "View_Help_2"     //bitmap id
//})
//DisplayTab( ULink; hbmp; hhlpbmp)
```

```
DLLCall(ULink; "LoadBitmap"; hhlpbmp1;WORD;{
     LOWORD(Link);  //hinstance
     "View_Help_2_1"     //bitmap id
})
DLLCall(ULink; "LoadBitmap"; hhlpbmp2;WORD;{
     LOWORD(Link);  //hinstance
     "View_Help_2_2"     //bitmap id
})
DLLCall(ULink; "LoadBitmap"; hhlpbmp3;WORD;{
     LOWORD(Link);  //hinstance
     "View_Help_2_3"     //bitmap id
})
DLLCall(ULink; "LoadBitmap"; hhlpbmp4;WORD;{
     LOWORD(Link);  //hinstance
     "View_Help_2_4"     //bitmap id
})

DisplayTab( ULink; hbmp1; hhlpbmp1)
DisplayTab( ULink; hbmp2; hhlpbmp2)
DisplayTab( ULink; hbmp3; hhlpbmp3)
DisplayTab( ULink; hbmp4; hhlpbmp4)

FreeWaitCursor(ULink; hCursor)
InhibitInput(Off!)
DialogDisplay ( dn; 1; DlgHelpCB@)

DoDlg := 1
While(DoDlg)
EndWhile

DialogDestroy ( dn)

RETURN
//************************************************************
```

```
//*************************************************************
//   ROUTINE NAME: DlgHelpCB@
//   INPUT VARIABLES:
//   RETURN VARIABLES:
//   USES:
//   DESCRIPTION:
//*************************************************************
LABEL(DlgHelpCB@)

If (InitDlg)
     InitDlg := 0
     hwnd := DlgHelpCB[4]
//   ShowWin(ULink; hWait; 0)
EndIf

// if SYS COMMAND
if(DlgHelpCB[5] = 274)
     DlgHelpCB[3] := 1
     //FreeHelp(hwnd; HelpFile) //Call(FreeHelp@)
     //DoDlg := 0
     //DialogUndisplay(dn;1)
     //DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hhlpbmp)})
endif


// HotSpots
if (SubStr(DlgHelpCB[3];1;1) = "h")
     hCursor := LoadWaitCursor(ULink; hPRWin)
     Switch (SubStr(DlgHelpCB[3];2;1))
       CaseOf "1":  SlideShowSlide ()
       CaseOf "2":  SlideShowOutline ()
       CaseOf "3":  SlideShowList ()
       CaseOf "4":  SlideShowSort ()
     EndSwitch
     FreeWaitCursor(ULink; hCursor)
endif

Switch(DlgHelpCB[3])
     CaseOf 1 :
       FreeHelp(hwnd; HelpFile) //Call(FreeHelp@)
       DoDlg := 0
       DialogUndisplay(dn;1)
       //DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hhlpbmp)})
       DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hhlpbmp1)})
       DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hhlpbmp2)})
       DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hhlpbmp3)})
       DLLCall(GLink;"DeleteObject"; ret:BOOL;{LOWORD(hhlpbmp4)})

     CaseOf "Help" :
       LoadHelp(hwnd; HelpFile; H_ITEM) //Call(LoadHelp@)

EndSwitch
```

```
RETURN
//*************************************************************
```

```
//*************************************************************
//   PROCEDURE SetText( ULink; hwnd; Text)
//   DESCRIPTION:
//*************************************************************
PROCEDURE SetText ( ULink; hwnd; Text)

     ShowWin( ULink; hwnd; 0)
     SetWinText( ULink; hwnd; Text)
     ShowWin( ULink; hwnd; 1)

EndProc
//*************************************************************
```

```
//*************************************************************
//   PROCEDURE DisplayTab( ULink; hctrl; hbmp)
//   DESCRIPTION:
//*************************************************************
PROCEDURE DisplayTab( ULink; hctrl; hbmp)

     DllCall(ULink;"SendMessage";ret:DWORD;{
        LOWORD(hctrl);      //hwnd of the control
        LOWORD(1024+900); //BMPM_SETHBITMAP message
        LOWORD(hbmp);       //handle to bitmap
        0                   //instance
     })

EndProc
//*************************************************************
```

```
//*************************************************************
//   PROCEDURE: DisplayText( ULink; htext)
//   DESCRIPTION:
//*************************************************************
PROCEDURE DisplayText( ULink; htext)

     ShowWin( ULink; htext; 0)
     ShowWin( ULink; htext; 1)

EndProc
//*************************************************************
```

```
//*********************************************************
//   ROUTINE NAME: WaitMsgDisply@
//   INPUT VARIABLES: None
//   RETURN VARIABLES: None
//   USES: Nothing
//   DESCRIPTION: Initializes macro wait messages.
//*********************************************************
Label(WaitMsgDisplay@)
// Include the following four lines for each wait message desired.
WaitDlgId:="Wait1"
WaitTitle:= AppName
WaitMessage:="Loading..."
DialogDefine(WaitDlgId;50;50;140;50;16;WaitTitle)
DialogAddText(WaitDlgId;"WaitText";8;10;124;20;1+4;WaitMessage)
DialogDisplay(WaitDlgId;0;WaitDlgCallBack@)
Return

Label(WaitDlgCallBack@)
Return
//*********************************************************
```

```
//*********************************************************
//   ROUTINE NAME:  WaitMsgDestroy@
//   INPUT VARIABLES:  WaitDlgId
//   OUTPUT VARIABLES:  None
//   USES:  Nothing
//   DESCRIPTION:   Hides a previously defined wait message dialog.
//*********************************************************
Label(WaitMsgDestroy@)
DialogUndisplay(WaitDlgId;1)
DialogDestroy(WaitDlgId)
Return
//*********************************************************
```

```
//*********************************************************
//   PROCEDURE:  LoadHelp(hwnd; HelpFile; H_ITEM)
//   DESCRIPTION:
//*********************************************************
PROCEDURE LoadHelp(hwnd; HelpFile; H_ITEM)

ULink := DLLLoad("USER")
HELP_CONTEXT := 1
H_ITEM := 1137

DLLCall( ULink; "WinHelp"; Ret:BOOL; {
     LOWORD(hwnd);
     ADDRESS( HelpFile);
     LOWORD(HELP_CONTEXT);
     H_ITEM
})

DLLFree(ULink)
//SLink:=DLLLoad("shwin20.dll")
//DLLCall(SLink;"whlpHelpInvoke";;VOID;{471})
//DLLFree(SLink)

ENDPROC
//*********************************************************
```

```
//*********************************************************
//   PROCEDURE:  FreeHelp(hwnd; HelpFile)
//   DESCRIPTION:
//*********************************************************
PROCEDURE FreeHelp(hwnd; HelpFile)

ULink := DLLLoad("USER")
HELP_QUIT := 2

DLLCall( ULink; "WinHelp"; Ret:BOOL; {
     LOWORD(hwnd);
     ADDRESS( HelpFile);
     LOWORD(HELP_QUIT);
     0
})

DLLFree(ULink)

ENDPROC
//*********************************************************
```

```
//*********************************************************
//   PROCEDURE: ShowWin ( ULink; hwnd; OnOff)
//   DESCRIPTION:
//*********************************************************
PROCEDURE ShowWin( ULink; hwnd; OnOff)

DLLCall(ULink;"ShowWindow";Ret:BOOL;{
     LOWORD(hwnd);
     LOWORD(OnOff)
})

EndProc
//*********************************************************
```

```
//*********************************************************
//   PROCEDURE: SetWinText( ULink; hwnd; Text)
//   DESCRIPTION:
//*********************************************************
PROCEDURE SetWinText( ULink; hwnd; Text)

     DLLCall( ULink; "SetWindowText"; ret:WORD;{
        LOWORD(hwnd); ADDRESS(Text)})

ENDPROC
//*********************************************************
```

```
//*********************************************************
//   FUNCTION: SelectFont( FontName, FontSize)
//   RETURN: handle to font
//*********************************************************
FUNCTION SelectFont( FontName; FontSize)
GLink := DLLLoad("GDI")

DLLCall(GLink;"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);
     FontName
})

DLLFree(GLink)
RETURN (hFont)
ENDFUNC
//*********************************************************
```

```
//*************************************************************
//   FUNCTION: SelectBoldFont( FontName, FontSize)
//   RETURN: handle to font
//*************************************************************
FUNCTION SelectBoldFont( FontName; FontSize)
GLink := DLLLoad("GDI")

DLLCall(GLink;"CreateFont"; hFont;WORD;{
     LOWORD(FontSize);
     LOWORD(0);LOWORD(0);LOWORD(0);LOWORD(700);
     LOWORD(0);LOWORD(0);LOWORD(0);LOWORD(0);
     LOWORD(0);LOWORD(0);LOWORD(0);LOWORD(0);
     FontName
})

DLLFree(GLink)
RETURN (hFont)
ENDFUNC
//*************************************************************
```

```
//*************************************************************
//   PROCEDURE: SetCtrlFont( DlgID, CtrlID, hFont)
//*************************************************************
PROCEDURE SetCtrlFont( DlgID; CtrlID; hFont)

ULink := DLLLoad ("USER")
DialogHandle(hwnd; DlgID; CtrlID)

DLLCall(ULink;"SendMessage";ret;DWORD;{
     LOWORD(hwnd);
     LOWORD(48);  //WM_SETFONT
     LOWORD(hFont);
     1
})

//DLLCall(GLink;"DeleteObject";Ret;WORD;{LOWORD(hFont)})

DLLFree (ULink)

ENDPROC
//*************************************************************
```

```
//************************************************************
//   PROCEDURE: FreeFont( hFont)
//************************************************************
PROCEDURE FreeFont( hFont)

GLink := DLLLoad ("GDI")

DLLCall(GLink;"DeleteObject";Ret;WORD;{LOWORD(hFont)})

DLLFree (GLink)

ENDPROC
//************************************************************
```

```
//*********************************************************
//   PROCEDURE: SetFocus ( ULink; hwnd)
//   DESCRIPTION:
//*********************************************************
PROCEDURE SetFocus ( ULink; hwnd)

DLLCall( ULink; "SetFocus"; Ret:WORD; {
     LOWORD(hwnd)
})

ENDPROC
//*********************************************************
```

```
//*****************************************************
//   FUNCTION: LoadWaitCursor (ULink; hwnd)
//   RETURN VALUE: hCursor
//   DESCRIPTION: Load wait mouse cursor.
//*****************************************************
FUNCTION LoadWaitCursor (ULink; hwnd)
IDC_WAIT := 32514

DLLCall(ULink;"SetCapture"; ret;WORD;{ LOWORD(hwnd)})
DLLCall(ULink;"LoadCursor"; hWaitCursor;WORD;{ LOWORD(0); IDC_WAIT})
DLLCall(ULink;"SetCursor" ; hCurCursor;WORD;{LOWORD(hWaitCursor)})

RETURN (hCurCursor)
ENDFUNC
//*****************************************************
```

```
//*****************************************************
//   PROCEDURE: FreeWaitCursor (ULink; hCursor)
//   DESCRIPTION: Free wait mouse cursor, and load original cursor.
//*****************************************************
PROCEDURE FreeWaitCursor (ULink; hCursor)

DLLCall(ULink;"SetCursor" ; hCursor;WORD;{LOWORD(hCursor)})
DLLCall(ULink;"ReleaseCapture"; ;VOID;{ })

ENDPROC
//*****************************************************
```

```
//*****************************************************
//   LABEL: SlideTemplateError@
//*****************************************************
LABEL(SlideTemplateError@)
RETURN
//*****************************************************
```