```
//*********************************************************
//        MACRO: PROMPTS.WCM
//        PURPOSE: Creates and edits the prompts inside of an automated template. Use when
editing or creating an automated template.
//*********************************************************
Application (A1; "WordPerfect"; Default; "UK")
PromptAbbr:="Template Prompts"    // Sets abbreviation name for prompts
MarkAbbr:="Template Bookmarks"  // Sets abbreviation name for bookmarks
ObjectFilename:="_autotmp.wpt"     // Name of sample automated template
ObjectName:="<dofiller>"              // Name of automation macro
MaxChars:=60                          // Sets maximum number of characters for a prompt
MaxPrompts:=12                        // Sets maximum number of prompts
NumLinks:=13                          // Sets number of possible address book links
Declare(Link[NumLinks])               // Holds link descriptions
Link[1]:="None"
Link[2]:="Name"
Link[3]:="Salutation"
Link[4]:="Title"
Link[5]:="Company"
Link[6]:="Address"
Link[7]:="Town"
Link[8]:="County"
Link[9]:="Postcode"
Link[10]:="Country"
Link[11]:="Telephone"
Link[12]:="Fax"
Link[13]:="Account/ID"

NumPersFields:=9                      // Sets number of personal information fields
Declare(PersField[NumPersFields])  // Holds personal field names
PersField[1]:="Name"
PersField[2]:="Title"
PersField[3]:="Company"
PersField[4]:="Address"
PersField[5]:="Town"
PersField[6]:="County"
PersField[7]:="Postcode"
PersField[8]:="Telephone"
PersField[9]:="Fax"
If(?BlockActive)
        SelectMode(Off!)
EndIf
DLLLoad(UserLink; "User")
If(Exists(ActivateCoach) = 2)
        UnderCoach:=True
        DLLLoad(WPCHLink;"WPCH61.DLL")
```

```
DllCall(USERLink;"GetFocus";hWndWP:Word;{})
BIFRead(
	Status:StatusVar;
	Group:"WordPerfect";
	Section:"Coaches";
	Item:"Font Name";
	Value:FontName;
	ExpectedItemType:AnsiString!;
	NameListType:Private!
)
If(StatusVar <= 0)
	FontName:="Helvetica"
Endif
BIFRead(
	Status:StatusVar;
	Group:"WordPerfect";
	Section:"Coaches";
	Item:"FontSize";
	Value:FontSize;
	ExpectedItemType:UnsignedWord!;
	NameListType:Private!
)
If(StatusVar <= 0)
	FontSize:=10
Endif
BIFRead(
	Status:StatusVar;
	Group:"WordPerfect";
	Section:"Coaches";
	Item:"Font Name (Title)";
	Value:FontNameTitle;
	ExpectedItemType:AnsiString!;
	NameListType:Private!
)
If(StatusVar <= 0)
	FontNameTitle:="Helvetica"
Endif
BIFRead(
	Status:StatusVar;
	Group:"WordPerfect";
	Section:"Coaches";
	Item:"FontSize (Title)";
	Value:FontSizeTitle;
	ExpectedItemType:UnsignedWord!;
	NameListType:Private!
)
```

```
	If(StatusVar <= 0)
		FontSizeTitle:=12
	Endif
	Call(DialogText@)
Else
	UnderCoach:=False
Endif
AllDone:=False
CancelMacro:=False
PastePersonal:=False

//*********************************************************
//	Program Begin
//*********************************************************
Call(IsDefault@)
Call(FindObject@)
Call(WaitMsgInit@)
DialogDefine("DlgMain";50;50;171;151;8+16;"Prompt Builder")
DialogDefine("DlgEdit";50;50;168;88;16;"")
DialogDefine("DlgPers";50;50;131;91;8+16;"Personal Fields")
OnNotFound(NotFound@)
Call(WaitMsgDisplay@)
Call(DefineDialogs@)
OnError(Error@)
Call(ParseAbbr@)

Call(WaitMsgHide@)

Display(On!)
InhibitInput(Off!)
DialogDisplay("DlgMain";"PromptList"; CBMain@)

MainActive:=0 Msg:=1 BttnFlg:=0 PromptActive:=0
Repeat
	If(UnderCoach)
		If(Msg = 1 and MainActive = 0)
			CoachMessageBoxClose(0)
			Call(CoachMessageBox@)
			DefBttn:=1
			MainActive:=1
		Endif
		Switch(BttnFlg)
		CaseOf Bttn1:
			If(Msg = 1 and MainActive = 1)
				CoachMessageBoxClose(0)
				Switch(DefBttn)
```

```
                CaseOf 1:
                        Msg:=3
                CaseOf 2:
                        NumItems:=LBGetCount(UserLink; hwndList)
                        If(NumItems = 0)
                                Msg:=8
                        Else
                                Msg:=6
                        Endif
                CaseOf 3:
                        Msg:=9

                EndSwitch
                Call(CoachMessageBox@)
                If(Msg = 8)
                        Msg:=1
                Endif
                MainActive:=0
        Else
                If(Msg = 8)
                        DialogDisplay("DlgMain";"PromptList"; CBMain@)
                        DialogDisplay("DlgMain"; "Personal"; CBMain@)
                Endif
                Msg:=1
        Endif
        BttnFlg:=0
CaseOf Bttn2:
        Call(QuitCoach@)
        BttnFlg:=0
CaseOf Bttn6:
        TmpMsg:=Msg
        Msg:=2
        Call(CoachMessageBox@)
        Msg:=TmpMsg
        BttnFlg:=0
CaseOf "RadBttn1":
        DefBttn:=1
        BttnFlg:=0
CaseOf "RadBttn2":
        DefBttn:=2
        BttnFlg:=0
CaseOf "RadBttn3":
        DefBttn:=3
        BttnFlg:=0

EndSwitch
```

```
        Endif
        If(UnderCoach)
                If(Msg = 6)
                        DllCall(USERLink;"GetFocus";hCurWnd:Word;{})
                        If(hCurWnd = hWndWP and PromptActive = 0)
                                CoachMessageBoxClose(0)
                                TmpMsg:=Msg
                                Msg:=7
                                Call(CoachMessageBox@)
                                Msg:=TmpMsg
                                PromptActive:=1
                        Else
                                If(hCurWnd <> hWndWP and PromptActive = 1)
                                        CoachMessageBoxClose(0)
                                        Call(CoachMessageBox@)
                                        PromptActive:=0
                                Endif
                        Endif
                Endif
        Endif
        If(PastePersonal)
                CoachMessageBoxClose(0)
                DialogUndisplay("DlgMain"; "Personal")
                Call(Personal@)
                PastePersonal:=False
                DialogDisplay("DlgMain"; "Personal"; CBMain@)
                Msg:=1 MainActive:=0
        EndIf
Until(AllDone)
DialogUndisplay("DlgMain"; "OK")
InhibitInput(On!)
Display(Off!)

If(Not CancelMacro)
        QuickMarkExists:=CheckQuickMark()
        Call(WaitMsgDisplay@)
        Call(MakePromptAbbr@)
        Call(BuildBookmarks@)
        Call(MakeBookmarkAbbr@)
        TemplateCopyObject(ObjectFile; Macro!; ObjectName)
        TemplateSetAssociation(PostNew!; Macro!; ObjectName)
        If(Not QuickMarkExists)
                PosDocTop()
                QuickMarkSet()
        EndIf
EndIf
```

```
Label(Stop@)
DllFree(UserLink)
If(UnderCoach)
	DllFree(WPCHLink)
Endif
Call(WaitMsgHide@)
Call(WaitMsgDestroy@)
Call(DestroyDialogs@)
Go(End@)
//*********************************************************
//	Program End
//*********************************************************

//*********************************************************
//	ROUTINE NAME:  IsDefault@
//	INPUT VARIABLES:  None
//	OUTPUT VARIABLES:  None
//	USES:
//	DESCRIPTION:  Warns user if playing macro in default template.
//*********************************************************
Label(IsDefault@)
If(ToUpper(?TemplateFile)=ToUpper(?CurrentTemplate))
	If(UnderCoach)
		Msg:=5
		Call(CoachMessageBox@)
	Endif
	MessageBox(; "Default Template"; "You cannot add prompts to your default template.
	To add prompts to a different template: edit the template then play PROMPTS.WCM
	again."; IconStop!)
	CoachMessageBoxClose(0)
	Go(End@)
EndIf
Return
//*********************************************************

//*********************************************************
//	ROUTINE NAME:  FindObject@
//	INPUT VARIABLES:  ObjectFile
//	OUTPUT VARIABLES:  ObjectFile
//	USES:  FindFile
//	DESCRIPTION:  Locates _autotmp.wpt.
//*********************************************************
Label(FindObject@)
ObjectFile:=FindFile(ObjectFilename)
If(ObjectFile="")
```

```
        MessageBox(; ObjectFilename+" Not Found"; "The template named ^0 was not found in
        either of your template directories.  Make certain that ^0 is in your template or
        supplementary template directory, then play PROMPTS.WCM again."; IconStop! |
        HasParameters!; ObjectFilename)
        Go(End@)
EndIf
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  ParseAbbr@
//      INPUT VARIABLES:  PromptAbbr
//      OUTPUT VARIABLES:  None
//      USES:  Nothing
//      DESCRIPTION:  Extracts prompts from abbreviation and stores in listbox.
//*********************************************************
Label(ParseAbbr@)
NumOfPrompts:=0
GetData(NumOfAbbrs;Abbreviation!;Count!;0)
If(NumOfAbbrs=0)
        Return
EndIf
// For the number of abbreviations,
For(CurrAbbrNum;1;CurrAbbrNum<=NumOfAbbrs;CurrAbbrNum+1)
        GetData(CurrAbbrName;Abbreviation!;Name!;0;CurrAbbrNum)
        If(CurrAbbrName=PromptAbbr)
                // Get contents of Abbreviation
                GetData(PullString;Abbreviation!;Data!;0;CurrAbbrNum)
                If(PullString="")
                        Go(StopParse@)
                Else
                        Go(ParseLoop@)
                EndIf
        EndIf
EndFor
Go(StopParse@)
Label(ParseLoop@)
DelPos:=StrPos(PullString;"|")
// Add the field to the listbox
NumOfPrompts:=NumOfPrompts+1
LBInsertWPString(UserLink; hwndList; NumOfPrompts-1; SubStr(PullString;1;DelPos-1))
// Cut assigned field from main string
Len:=StrLen(PullString)
PullString:=SubStr(PullString;DelPos+1;Len-DelPos)
// Assign the link number to the listbox
DelPos:=StrPos(PullString;"|")
```

```
If(DelPos=0)   // If last field to parse,
        LBSetItemData(UserLink; hwndList; NumOfPrompts-1; StrNum(PullString))
        // highlight first prompt
        Go(StopParse@)
Else
        LBSetItemData(UserLink; hwndList; NumOfPrompts-1; StrNum(SubStr(PullString; 1;
        DelPos-1)))
        Len:=StrLen(PullString)
        PullString:=SubStr(PullString;DelPos+1;Len-DelPos)
        Go(ParseLoop@)
EndIf
Label(StopParse@)
Discard(NumOfAbbrs;CurrAbbrNum;CurrAbbrName;PullString;DelPos;Len)
Return
//*********************************************************

//*********************************************************
//        ROUTINE NAME:  DefineDialogs@
//        INPUT VARIABLES:  NumOfPrompts; NewPrompt; MaxChars; NewLink; NumLinks;
Link[]; PersField[]; NumPersFields; PersFld
//        OUTPUT VARIABLES:  hwndList
//        USES:  Nothing
//        DESCRIPTION:  Defines all macro dialogs.
//*********************************************************
Label(DefineDialogs@)
// Main Dialog
DialogAddPushButton("DlgMain";"OK";116;8;43;13;0;"OK")
DialogAddPushButton("DlgMain";"Cancel";116;24;43;13;0;"Cancel")
DialogAddPushButton("DlgMain";"Add";116;45;43;13;1;"&Add...")
DialogAddPushButton("DlgMain";"Paste";116;61;43;13;0;"&Paste")
DialogAddPushButton("DlgMain";"Edit";116;77;43;13;0;"&Edit...")
DialogAddPushButton("DlgMain";"Delete";116;93;43;13;0;"&Delete")
DialogAddPushButton("DlgMain";"Personal";116;114;43;13;0;"Pe&rsonal...")
DialogAddText("DlgMain";"T1";8;8;100;10;1;"&Template Prompts:")
DialogAddControl("DlgMain";"PromptList";8;19;100;48;"WPListbox20";0A10000x;"";
CurrPrompt; 0)
DialogAddPushButton("DlgMain"; "MoveUp"; 8; 75; 48; 13; 0; "Move &Up")
DialogAddPushButton("DlgMain"; "MoveDn"; 60; 75; 48; 13; 0; "Move Do&wn")
DialogAddText("DlgMain";"Help";8;94;100;35;1;"Add the prompts your template will display.
Paste the prompts where you want the responses to appear.")
DialogHandle(hwndList; "DlgMain"; "PromptList")
DialogHandle(hwndEdit; "DlgMain"; "Edit")
DialogHandle(hwndPaste; "DlgMain"; "Paste")
DialogHandle(hwndDelete; "DlgMain"; "Delete")
DialogHandle(hwndMoveUp; "DlgMain"; "MoveUp")
DialogHandle(hwndMoveDn; "DlgMain"; "MoveDn")
```

```
DialogHandle(hwndDlgMain; "DlgMain")
// Edit Dialog
DialogAddText("DlgEdit";"T1";8;8;100;10;1;"&Prompt:")
DialogAddEditBox("DlgEdit";"EB1";8;19;100;13;32;NewPrompt;MaxChars)
DialogAddText("DlgEdit";"T2";8;38;100;10;1;"&Link to Address Book Field:")
DialogAddPopUpButton("DlgEdit";"Pop1";8;49;68;13;NewLink)
For(Count;1;Count<=NumLinks;Count+1)
        DialogAddListItem("DlgEdit";"Pop1";Link[Count])
EndFor
DialogAddPushButton("DlgEdit";"OK";116;8;38;13;1;"OK")
DialogAddPushButton("DlgEdit";"Cancel";116;24;38;13;0;"Cancel")
DialogHandle(hwndDlgEdit; "DlgEdit")
// Personal Dialog
DialogAddListBox("DlgPers";"PersList";8;8;65;65;2;PersFld)
For(Count;1;Count<=NumPersFields;Count+1)
        DialogAddListItem("DlgPers";"PersList";PersField[Count])
EndFor
DialogAddPushButton("DlgPers";"Paste";81;8;38;13;1;"&Paste")
DialogAddPushButton("DlgPers";"Close";81;24;38;13;0;"&Close")
DialogHandle(hwndPersList; "DlgPers"; "PersList")
DialogHandle(hwndPersPaste; "DlgPers"; "Paste")
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  CBMain@
//      INPUT VARIABLES:  None
//      OUTPUT VARIABLES:  AllDone; PastePersonal; CancelMacro
//      USES:  LB... Functions
//      DESCRIPTION:  Callback routine for main dialog.
//*********************************************************
Label(CBMain@)
Switch(CBMain[3])
        CaseOf "":
                Switch(CBMain[5])
                        CaseOf 6:       // WM_ACTIVATE message
                                CurSel:=LBGetCurSel(UserLink; hwndList)
                                NumItems:=LBGetCount(UserLink; hwndList)
                                If((CurSel=-1) And (NumItems>0))
                                        LBSetCurSel(UserLink; hwndList; 0)
                                EndIf
                                EnableWindow(UserLink; hwndEdit; NumItems>0)
                                EnableWindow(UserLink; hwndDelete; NumItems>0)
                                EnableWindow(UserLink; hwndPaste; NumItems>0)
                                EnableWindow(UserLink; hwndMoveUp; NumItems>1)
                                EnableWindow(UserLink; hwndMoveDn; NumItems>1)
```

```
                CaseOf 274:    // WM_SYSCOMMAND
                        CoachMessageBoxClose(0)
                        AllDone:=True
                        CancelMacro:=True
        EndSwitch
CaseOf "Add":
        CoachMessageBoxClose(0)
        If(UnderCoach)
                DialogUndisplay("DlgMain";"Add")
                Call(Add@)
                DialogDisplay("DlgMain";"Add"; CBMain@)
        Else
                EnableWindow(UserLink; hwndDlgMain; 0)
                Call(Add@)
                EnableWindow(UserLink; hwndDlgMain; 1)
        EndIf
        Msg:=1 MainActive:=0
CaseOf "Edit":
        CoachMessageBoxClose(0)
        If(UnderCoach)
                DialogUndisplay("DlgMain";"Edit")
                Call(Edit@)
                DialogDisplay("DlgMain";"Edit"; CBMain@)
        Else
                EnableWindow(UserLink; hwndDlgMain; 0)
                Call(Edit@)
                EnableWindow(UserLink; hwndDlgMain; 1)
        EndIf
        Msg:=1 MainActive:=0
CaseOf "Delete":
        Call(Delete@)
CaseOf "Paste":
        Call(Paste@)
CaseOf "Personal":
        CoachMessageBoxClose(0)
        PastePersonal:=True
        Msg:=1 MainActive:=0
CaseOf "OK":
        CoachMessageBoxClose(0)
        AllDone:=True
CaseOf "Cancel":
        CoachMessageBoxClose(0)
        AllDone:=True
        CancelMacro:=True
CaseOf "MoveDn": MoveItemDown(UserLink; hwndList)
CaseOf "MoveUp": MoveItemUp(UserLink; hwndList)
```

```
        Default: Return
EndSwitch
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  CBPers@
//      INPUT VARIABLES:  None
//      OUTPUT VARIABLES:  DonePasting
//      USES:  LBGetCurSel; LBGetText
//      DESCRIPTION:  Callback routine for personal dialog.
//*********************************************************
Label(CBPers@)
Switch(CBPers[3])
        CaseOf "Close":
                DonePasting:=True
        CaseOf "Paste":
                Type("<"+LBGetText(UserLink; hwndPersList; LBGetCurSel(UserLink;
                hwndPersList))+">")
EndSwitch
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  Add@
//      INPUT VARIABLES:  MaxPrompts
//      OUTPUT VARIABLES:  NewPrompt
//      USES:  ErrorChecks@; GetLinkNum@; LBGetCount
//      DESCRIPTION:  Adds new prompt to prompt list.
//*********************************************************
Label(Add@)
If(LBGetCount(UserLink; hwndList)>=MaxPrompts)
        MessageBox(; "Too Many Prompts"; "You cannot add more than ^0 prompts.";
        IconExclamation! | HasParameters!; MaxPrompts)
        Return
EndIf
If(?BlockActive)
        NewPrompt:=?SelectedText
Else
        NewPrompt:=""
EndIf
NewLink:=Link[1]     // Set initial link to "none"
SetWindowText(UserLink; hwndDlgEdit; "Add Template Prompt")
Label(DispAdd@)
If(UnderCoach)
        Msg:=4
```

```
        Call(CoachMessageBox@)
Endif
DialogDisplay("DlgEdit";"EB1")
CoachMessageBoxClose(0)
If(MacroDialogResult="OK")
        Call(ErrorChecks@)
        If(ErrorExists)
                Go(DispAdd@)
        EndIf
        If(?BlockActive)
                Type("["+NewPrompt+"]")
        EndIf
        Call(GetLinkNum@)
        Index:=LBGetCount(UserLink; hwndList)
        LBInsertWPString(UserLink; hwndList; Index; NewPrompt)
        LBSetItemData(UserLink; hwndList; Index; LinkNum)
        LBSetCurSel(UserLink; hwndList; Index)
        GotEvent:=1
EndIf
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  Paste@
//      INPUT VARIABLES:
//      OUTPUT VARIABLES:  None
//      USES:  CheckSelection; LBGetCurSel
//      DESCRIPTION:  Pastes highlighted prompt in document.
//*********************************************************
Label(Paste@)
If(Not CheckSelection(UserLink; hwndList))
        Return
EndIf
Index:=LBGetCurSel(UserLink; hwndList)
Type("["+LBGetWPText(UserLink; hwndList; Index)+"]")
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  Edit@
//      INPUT VARIABLES:
//      OUTPUT VARIABLES:
//      USES:  CheckSelection; LBGetCurSel; ErrorChecks@; UpdateDoc; LBDeleteString;
LBInsertWPString; GetLinkNum@
//      DESCRIPTION:  Edits the highlighted prompt.
//*********************************************************
```

```
Label(Edit@)
If(Not CheckSelection(UserLink; hwndList))
	Return
EndIf
Index:=LBGetCurSel(UserLink; hwndList)
NewLink:=Link[LBGetItemData(UserLink; hwndList; Index)+1]
NewPrompt:=LBGetWPText(UserLink; hwndList; Index)
Label(DispEdit@)
SetWindowText(UserLink; hwndDlgEdit; "Edit Template Prompt")
If(UnderCoach)
	Msg:=4
	Call(CoachMessageBox@)
Endif
DialogDisplay("DlgEdit";"EB1")
CoachMessageBoxClose(0)
If(MacroDialogResult="OK")
	If(NewPrompt=LBGetWPText(UserLink; hwndList; Index))
		If(NewLink=Link[LBGetItemData(UserLink; hwndList; Index)+1])
			Return
		Else
			Go(ChangeLink@)
		EndIf
	EndIf
	Call(ErrorChecks@)
	If(ErrorExists)
		Go(DispEdit@)
	EndIf
	UpdateDoc(LBGetWPText(UserLink; hwndList; Index); NewPrompt)
	OnNotFound(NotFound@)
	LBDeleteString(UserLink; hwndList; Index)
	LBInsertWPString(UserLink; hwndList; Index; NewPrompt)
	LBSetCurSel(UserLink; hwndList; Index)
	Label(ChangeLink@)
	Call(GetLinkNum@)
	LBSetItemData(UserLink; hwndList; Index; LinkNum)
EndIf
Return
//*********************************************************

//*********************************************************
//	ROUTINE NAME:  Delete@
//	INPUT VARIABLES:  None
//	OUTPUT VARIABLES:  None
//	USES:  CheckSelection; UpdateDoc; LBGetCurSel; LBDeleteString
//	DESCRIPTION:  Deletes the highlighted prompt.
//*********************************************************
```

```
Label(Delete@)
If(Not CheckSelection(UserLink; hwndList))
	Return
EndIf
Index:=LBGetCurSel(UserLink; hwndList)
MessageBox(MsgBoxResult; "Confirm Deletion"; "Are you sure you wish to delete ""^0""?";
IconQuestion! | YesNo! | HasParameters!; LBGetWPText(UserLink; hwndList; Index))
If(MsgBoxResult=6)  //YES
	UpdateDoc(LBGetWPText(UserLink; hwndList; Index); "")
	OnNotFound(NotFound@)
	// Determine which prompt to highlight next
	If(Index=LBGetCount(UserLink; hwndList)-1)
		NewIndex:=Index-1
	Else
		NewIndex:=Index
	EndIf
	LBDeleteString(UserLink; hwndList; Index)
	LBSetCurSel(UserLink; hwndList; NewIndex)
EndIf
Return
//*********************************************************

//*********************************************************
//	ROUTINE NAME:  Personal@
//	INPUT VARIABLES:  PersField[]
//	OUTPUT VARIABLES:  None
//	USES:  Nothing
//	DESCRIPTION:  Displays personal field dialog.
//*********************************************************
Label(Personal@)
PersFld:=PersField[1]
If(UnderCoach)
	Msg:=11
	Call(CoachMessageBox@)
Endif
DonePasting:=False
DialogDisplay("DlgPers";"PersList"; CBPers@)
Repeat
	If(UnderCoach)
		Switch(BttnFlg)
		CaseOf Bttn2:
			Call(QuitCoach@)
			BttnFlg:=0

		EndSwitch
	Endif
```

```
        If(UnderCoach)
                If(Msg = 11)
                        DllCall(USERLink;"GetFocus";hCurWnd:Word;{})
                        If(hCurWnd = hWndWP and PromptActive = 0)
                                CoachMessageBoxClose(0)
                                TmpMsg:=Msg
                                Msg:=7
                                Call(CoachMessageBox@)
                                Msg:=TmpMsg
                                PromptActive:=1
                        Else
                                If(hCurWnd <> hWndWP and PromptActive = 1)
                                        CoachMessageBoxClose(0)
                                        Call(CoachMessageBox@)
                                        PromptActive:=0
                                Endif
                        Endif
                Endif
        Endif
Until(DonePasting)
DialogUndisplay("DlgPers";"Close")
CoachMessageBoxClose(0)
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  MakePromptAbbr@
//      INPUT VARIABLES:  PromptAbbr
//      OUTPUT VARIABLES:  None
//      USES:  Nothing
//      DESCRIPTION:  Creates the "Template Prompts" abbreviation.
//*********************************************************
Label(MakePromptAbbr@)
OnNotFound(SkipDelete@)
AbbreviationDelete(PromptAbbr;CurrentDoc!)
Label(SkipDelete@)
OnNotFound(NotFound@)
NumOfPrompts:=LBGetCount(UserLink; hwndList)
If(NumOfPrompts>0)
        AbbrString:=LBGetWPText(UserLink; hwndList; 0)+"|"+LBGetItemData(UserLink;
        hwndList; 0)
        For(Count;1;Count<NumOfPrompts;Count+1)
                AbbrString:=AbbrString+"|"+LBGetWPText(UserLink; hwndList; Count)
                +"|"+LBGetItemData(UserLink; hwndList; Count)
        EndFor
        AbbreviationCreate(PromptAbbr;CurrentDoc!;AbbrString)
```

```
EndIf
Return
//*********************************************************

//*********************************************************
//        ROUTINE NAME:  BuildBookmarks@
//        INPUT VARIABLES:  NumPersFields; MarkAbbr
//        OUTPUT VARIABLES:  BookmarkString
//        USES:  Nothing
//        DESCRIPTION:  Builds bookmarks around prompts in document.
//*********************************************************
Label(BuildBookmarks@)
If(?DocBlank)
        Go(Stop@)
EndIf
// Create bookmarks for prompts
MarkNumber:=0
BookmarkString:=""
For(Count;1;Count<=NumPersFields;Count+1)
        PosDocVeryTop()
        While(True)
                OnNotFound(NoMorePersonal@)
                SearchString("<"+PersField[Count]+">")
                MatchSelection()
                SearchNext(Extended!)
                MarkNumber:=MarkNumber+1
                BookmarkCreate(MarkNumber)
                BookmarkString:=BookmarkString+Count+"|"
                Next
                Label(NoMorePersonal@)
                OnNotFound(NotFound@)
                While(?Substructure)
                        SubDoc:=?CurrentSubDoc
                        SubstructureExit()
                        If((SubDoc=10) Or (SubDoc=11))
                                BoxEnd(Save!)
                        EndIf
                EndWhile
                Break
        EndWhile
EndFor
If(BookmarkString="")
        BookmarkString:="||"
Else
        BookmarkString:=BookmarkString+"|"
EndIf
```

```
PromptFound:=False
For(Count;1;Count<=LBGetCount(UserLink; hwndList);Count+1)
	PosDocVeryTop()
	While(True)
		OnNotFound(NoMorePrompts@)
		SearchString("["+LBGetWPText(UserLink; hwndList; Count-1)+"]")
		MatchSelection ()
		SearchNext(Extended!)
		PromptFound:=True
		MarkNumber:=MarkNumber+1
		BookmarkCreate(MarkNumber)
		BookmarkString:=BookmarkString+Count+"|"
		Next
		Label(NoMorePrompts@)
		OnNotFound(NotFound@)
		While(?Substructure)
			SubDoc:=?CurrentSubDoc
			SubstructureExit()
			If((SubDoc=10) Or (SubDoc=11))
				BoxEnd(Save!)
			EndIf
		EndWhile
		Break
	EndWhile
EndFor
If(PromptFound)
	BookmarkString:=SubStr(BookmarkString;1;StrLen(BookmarkString)-1)
EndIf
Return
//*********************************************************

//*********************************************************
//	ROUTINE NAME:  MakeBookmarkAbbr@
//	INPUT VARIABLES:  BookmarkString; MarkAbbr
//	OUTPUT VARIABLES:  None
//	USES:  Nothing
//	DESCRIPTION:  Creates the "Bookmarks" abbreviation.
//*********************************************************
Label(MakeBookmarkAbbr@)
OnNotFound(DoNotDelete@)
AbbreviationDelete(MarkAbbr;CurrentDoc!)
Label(DoNotDelete@)
OnNotFound(NotFound@)
AbbreviationCreate(MarkAbbr;CurrentDoc!;BookmarkString)
Discard(BookmarkString)
Return
```

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

//*********************************************************
//        ROUTINE NAME:  DestroyDialogs@
//        INPUT VARIABLES:  None
//        OUTPUT VARIABLES:  None
//        USES:  Nothing
//        DESCRIPTION:  Destroys all macro dialogs.
//*********************************************************
Label(DestroyDialogs@)
DialogDestroy("DlgMain")
DialogDestroy("DlgEdit")
DialogDestroy("DlgPers")
Return
//*********************************************************

//*********************************************************
//        ROUTINE NAME:  ErrorChecks@
//        INPUT VARIABLES:  NewPrompt; MaxChars
//        OUTPUT VARIABLES:  None
//        USES:
//        DESCRIPTION:  Performs various error checks on new prompts.
//*********************************************************
Label(ErrorChecks@)
ErrorExists:=False
// Blank Entry
If(NewPrompt="")
        MessageBox(; "No Prompt Specified"; "You must enter text before pressing OK.";
        IconExclamation!)
        ErrorExists:=True
        Return
EndIf
// Invalid Character(s)
DelPos:=StrPos(NewPrompt;"|")
If(DelPos<>0)
        MessageBox(; "Illegal Character"; "You cannot include the pipe character (vertical bar)
        in the prompt."; IconExclamation!)
        ErrorExists:=True
        Return
EndIf
// Too Many Characters
PromptLen:=StrLen(NewPrompt)
If(PromptLen>MaxChars)
        MessageBox(; "Too Many Characters"; "Your prompt cannot be longer than ^0
        characters."; IconExclamation! | HasParameters!; MaxChars)
        ErrorExists:=True
```

```
        Return
EndIf
// Duplicate Prompts
For(Count;0;Count<=LBGetCount(UserLink; hwndList)-1;Count+1)
        If(ToUpper(NewPrompt)=ToUpper(LBGetWPText(UserLink; hwndList; Count)))
                MessageBox(; "Duplicate Entry"; "This prompt already exists.  Please choose
                another name."; IconExclamation!)
                ErrorExists:=True
                Return
        EndIf
EndFor
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  GetLinkNum@
//      INPUT VARIABLES:  NumLinks; NewLink; Link[]
//      OUTPUT VARIABLES:  LinkNum
//      USES:  Nothing
//      DESCRIPTION:  Returns number of the given address book link.
//*********************************************************
Label(GetLinkNum@)
For(Count;1;Count<=NumLinks;Count+1)
        If(NewLink=Link[Count])
                LinkNum:=Count-1
                Return
        EndIf
EndFor
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME: WaitMsgInit@
//      INPUT VARIABLES: None
//      RETURN VARIABLES: None
//      USES:  Nothing
//      DESCRIPTION: Initializes macro wait messages.
//*********************************************************
Label(WaitMsgInit@)
WaitDlgId:="WaitOpen"
WaitTitle:="Prompt Builder"
WaitMessage:="Preparing template..."
DialogDefine(WaitDlgId;50;50;140;50;16+32;WaitTitle)
DialogAddText(WaitDlgId;"WaitText";8;12;124;10;1+4;WaitMessage)
Return
//*********************************************************
```

```
//*********************************************************
//      ROUTINE NAME:  WaitMsgDisplay@
//      INPUT VARIABLES:  WaitDlgId
//      OUTPUT VARIABLES:  None
//      USES:  Nothing
//      DESCRIPTION:  Displays a previously defined wait message dialog.
//*********************************************************
Label(WaitMsgDisplay@)
DialogDisplay(WaitDlgId;0;WaitDlgCallBack@)
Return

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

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

//*********************************************************
//      ROUTINE NAME:  WaitMsgDestroy@
//      INPUT VARIABLES:  WaitDlgId
//      OUTPUT VARIABLES:  None
//      USES:  Nothing
//      DESCRIPTION:  Destroys a previously defined wait message dialog. Hide the dialog
with WaitMsgHide@ before destroying it.
//*********************************************************
Label(WaitMsgDestroy@)
DialogDestroy(WaitDlgId)
Return
//*********************************************************

//*********************************************************
//      ROUTINE NAME:  Error@
//      INPUT VARIABLES:  None
//      OUTPUT VARIABLES:  None
//      USES:
```

```
//        DESCRIPTION:  Displays unexpected error message.
//*************************************************************
Label(Error@)
MessageBox(; "Unexpected Error Condition"; "An unexpected error has occurred."; IconStop!)
Go(Stop@)
//*************************************************************

//*************************************************************
//        ROUTINE NAME:  NotFound@
//        INPUT VARIABLES:  None
//        OUTPUT VARIABLES:  None
//        USES:
//        DESCRIPTION:  Displays unexpected not found message.
//*************************************************************
Label(NotFound@)
MessageBox(; "Unexpected ""Not Found"" Condition"; "An unexpected not found has
occurred."; IconStop!)
Go(Stop@)
//*************************************************************

//*************************************************************
//        PROCEDURE NAME:  UpdateDoc
//        INPUT VARIABLES:  OldText; NewText
//        OUTPUT VARIABLES:  None
//        USES:  Nothing
//        DESCRIPTION:  Updates the prompt fields in the document.
//*************************************************************
Procedure UpdateDoc(OldText; NewText)
TempMark:="_prompt.wcm"            // Sets name of temporary bookmark
Display(Off!)
BookmarkCreate(TempMark)
PosDocVeryTop()
OnNotFound(RepReturn@)
SearchString("["+OldText+"]")
If(NewText<>"")
        ReplaceString("["+NewText+"]")
Else
        ReplaceString("")
EndIf
ReplaceForward(Extended!)
While(?Substructure)
        SubDoc:=?CurrentSubDoc
        SubstructureExit()
        If((SubDoc=10) Or (SubDoc=11))
                BoxEnd(Save!)
        EndIf
```

```
EndWhile
Label(RepReturn@)
BookmarkFind(TempMark)
BookmarkDelete(TempMark)
Display(On!)
EndProc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  FindFile
//      INPUT:  Name of file to search for
//      OUTPUT:  Path of file or NULL string
//      USES: Nothing
//      DESCRIPTION:  Locates a file in the template directories.
//*********************************************************
Function FindFile(Name)
FileExists(FExists; Name)
If (FExists)
      Return(Name)
Else
      NewName:=?PathTemplate+Name
EndIf
FileExists(FExists; NewName)
If (FExists)
      Return(NewName)
Else
      NewName:=?PathTemplateSupplemental+Name
EndIf
FileExists(FExists; NewName)
If (FExists)
      Return(NewName)
EndIf
Return("")
EndFunc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  CheckQuickMark
//      INPUT:  Nothing
//      OUTPUT:  True if QuickMark exists, otherwise False
//      USES: Nothing
//      DESCRIPTION:  Checks on the existance of a QuickMark.
//*********************************************************
Function CheckQuickMark()
Existance:=False
GetData(Num;Bookmark!;Count!;CurrentDoc!)
```

```
If(Num=0)
	Return(Existance)
EndIf
For(Curr;1;Curr<=Num;Curr+1)
	GetData(CurrName;Bookmark!;Name!;CurrentDoc!;Curr)
	If(CurrName="QuickMark")
		Existance:=True
		Break
	EndIf
EndFor
Return(Existance)
EndFunc
//*********************************************************

//*********************************************************
//	PROCEDURE NAME:  MoveItemUp
//	INPUT:  Link; Hwnd; Index
//	OUTPUT:  None
//	USES:  LBGetCurSel; LBGetWPText; LBGetItemData; LBDeleteString;
LBInsertWPString; LBSetItemData; LBSetCurSel
//	DESCRIPTION:  Moves the highlighted item up one location.
//*********************************************************
Procedure MoveItemUp(Link; Hwnd)
Index:=LBGetCurSel(Link; Hwnd)
If (Index>0)
	MoveString:=LBGetWPText(Link; Hwnd; Index)
	MoveData:=LBGetItemData(Link; Hwnd; Index)
	LBDeleteString(Link; Hwnd; Index)
	Index:=Index-1
	LBInsertWPString(Link; Hwnd; Index; MoveString)
	LBSetItemData(Link; Hwnd; Index; MoveData)
	LBSetCurSel(Link; Hwnd; Index)
Else
	Beep
EndIf
EndProc
//*********************************************************

//*********************************************************
//	PROCEDURE NAME:  MoveItemDown
//	INPUT:  Link; Hwnd
//	OUTPUT:  None
//	USES:  LBGetCount; LBGetCurSel; LBGetWPText; LBGetItemData; LBDeleteString;
LBInsertWPString; LBSetItemData; LBSetCurSel
//	DESCRIPTION:  Moves hilighted item down one location.
//*********************************************************
```

```
Procedure MoveItemDown(Link; Hwnd)
ListCount:=LBGetCount(Link; Hwnd)
Index:=LBGetCurSel(Link; Hwnd)
If (Index<ListCount-1)
        MoveString:=LBGetWPText(Link; Hwnd; Index)
        MoveData:=LBGetItemData(Link; Hwnd; Index)
        LBDeleteString(Link; Hwnd; Index)
        Index:=Index+1
        LBInsertWPString(Link; Hwnd; Index; MoveString)
        LBSetItemData(Link; Hwnd; Index; MoveData)
        LBSetCurSel(Link; Hwnd; Index)
Else
        Beep
EndIf
EndProc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  CheckSelection
//      INPUT:  Link; Hwnd
//      OUTPUT:  True if an item is selected.
//      USES:  LBGetCurSel
//      DESCRIPTION:  Determines if an item is currently highlighted in the listbox.
//*********************************************************
Function CheckSelection(Link; Hwnd)
Highlight:=True
If(LBGetCurSel(Link; Hwnd)<0)
        MessageBox(; "No Prompt Selected"; "You must select an item before using this
        option."; IconExclamation!)
        HighLight:=False
EndIf
Return(HighLight)
EndFunc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  LBGetCurSel
//      INPUT:  Link; Hwnd
//      OUTPUT:  Index of highlighted item.
//      USES:  Nothing
//      DESCRIPTION:  Gets the index of the currently highlighted item or -1 if none is
highlighted.
//*********************************************************
Function LBGetCurSel(Link; Hwnd)
DLLCall(Link; "SendMessage"; nIndex:INTEGER; {LoWord(Hwnd); LoWord(1033);
LoWord(0); 0})
```

```
Return(nIndex)
EndFunc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  LBSetCurSel
//      INPUT:  Link; Hwnd; Index
//      OUTPUT:  None.
//      USES:  Nothing
//      DESCRIPTION:  Sets the selection in a listbox.
//*********************************************************
Procedure LBSetCurSel(Link; Hwnd; Index)
DLLCall(Link; "SendMessage"; nIndex:INTEGER; {LoWord(Hwnd); LoWord(1031);
LoWord(Index); 0})
EndProc
//*********************************************************

//*********************************************************
//      PROCEDURE NAME:  LBDeleteString
//      INPUT:   Link; Hwnd; Index
//      OUTPUT:  Number of items left in list or -1 if error.
//      USES:  Nothing
//      DESCRIPTION:  Deletes an item from a listbox.
//*********************************************************
Procedure LBDeleteString(Link; Hwnd; Index)
DLLCall(Link; "SendMessage"; nNum:INTEGER; {LoWord(Hwnd); LoWord(1027);
LoWord(Index); 0})
EndProc
//*********************************************************

//*********************************************************
//      PROCEDURE NAME:  LBInsertWPString
//      INPUT:  Link; Hwnd; Index; NewString
//      OUTPUT:  Position where string was inserted or -1 if error.
//      USES:  Nothing
//      DESCRIPTION:  Adds an item to a dialog list at the specified position.
//*********************************************************
Procedure LBInsertWPString(Link; Hwnd; Index; NewString)
DllCall(Link; "SendMessage"; Ret:INTEGER; {LoWord(Hwnd); LoWord(1026);
LoWord(Index); WPString(NewString)})
EndProc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  LBGetItemData
//      INPUT:  Link; Hwnd; Index
```

```
//        OUTPUT:  Value associated with specified list item.
//        USES:  Nothing
//        DESCRIPTION:  Returns the value associated with the specified list item.
//*********************************************************
Function LBGetItemData(Link; Hwnd; Index)
DLLCall(Link; "SendMessage"; dwData:DWORD; {LoWord(Hwnd); LoWord(1050);
LoWord(Index); 0})
Return(dwData)
EndFunc
//*********************************************************

//*********************************************************
//        PROCEDURE NAME:  LBSetItemData
//        INPUT:  Link; Hwnd; Index; Data
//        OUTPUT:  None
//        USES:  Nothing
//        DESCRIPTION:  Sets the value associated with the specified list item.
//*********************************************************
Procedure LBSetItemData(Link; Hwnd; Index; Data)
DllCall(Link; "SendMessage"; RetVal:INTEGER; {LoWord(Hwnd); LoWord(1051);
LoWord(Index); Data})
EndProc
//*********************************************************

//*********************************************************
//        FUNCTION NAME:  LBGetCount
//        INPUT:  Link; Hwnd
//        OUTPUT:  Number of items in list box.
//        USES:  Nothing
//        DESCRIPTION:  Gets the number of items in the list box.
//*********************************************************
Function LBGetCount(Link; Hwnd)
DllCall(Link; "SendMessage"; Count:INTEGER; {LoWord(Hwnd); LoWord(1036); LoWord(0);
0})
Return(Count)
EndFunc
//*********************************************************

//*********************************************************
//        FUNCTION NAME:  LBGetWPText
//        INPUT:  Link; Hwnd; Index
//        OUTPUT:  Text of specified list box item.
//        USES:  Nothing
//        DESCRIPTION:  Returns text of specified list item.
//*********************************************************
Function LBGetWPText(Link; Hwnd; Index)
```

```
DllCall(Link; "SendMessage"; Ret:INTEGER; {LoWord(Hwnd); LoWord(1034);
LoWord(Index); Address(WPString(NewString))})
Return(NewString)
EndFunc
//*********************************************************

//*********************************************************
//      FUNCTION NAME:  LBGetText
//      INPUT:  Link; Hwnd; Index
//      OUTPUT:  Text of specified list box item.
//      USES:  Nothing
//      DESCRIPTION:  Returns text of specified list item.
//*********************************************************
Function LBGetText(Link; Hwnd; Index)
DllCall(Link; "SendMessage"; Ret:INTEGER; {LoWord(Hwnd); LoWord(1034);
LoWord(Index); Address(AnsiString(NewString))})
Return(NewString)
EndFunc
//*********************************************************

//*********************************************************
//      PROCEDURE NAME:  SetWindowText
//      INPUT:  Link; Hwnd; NewString
//      OUTPUT:  None
//      USES:  Nothing
//      DESCRIPTION:  Sets the text of a specified window.
//*********************************************************
Procedure SetWindowText(Link; Hwnd; NewString)
DllCall(Link; "SetWindowText"; :Void; {LoWord(Hwnd); AnsiString(NewString)})
EndProc
//*********************************************************

//*********************************************************
//      PROCEDURE NAME:  EnableWindow
//      INPUT:  Link; Hwnd; Mode(1=enable, 0=disable)
//      OUTPUT:  None
//      USES:  Nothing
//      DESCRIPTION:  Enables or disables the specified window.
//*********************************************************
Procedure EnableWindow(Link; Hwnd; Mode)
DllCall(Link; "EnableWindow"; Ret:Bool; {LoWord(Hwnd); LoWord(Mode)})
EndProc
//*********************************************************

//*********************************************************
//      SUBROUTINE: QuitCoach
```

```
//      DESC:
//*************************************************************
LABEL(QuitCoach@)
        CoachMessageBox(
                0;ArrayMsg[10,1]; ;
                WPCHLink;100;0;0;255;0;Top!;;50;50;FontName;FontSize;;;
                Simple!;;Result;
                Bttn5;Button!;Bttn5;
                Bttn4;Button!;Bttn4)
        If(Result = Bttn5)
                Discard(ActivateCoach)
                UnderCoach:=False
                CoachMessageBoxClose(0)
        Endif
RETURN

//*************************************************************
//      SUBROUTINE:  CoachMessageBox
//      DESC: Constructs a coach message box
//*************************************************************
LABEL(CoachMessageBox@)
        Switch(ArrayMsg[Msg,1])
        CaseOf 1:
                ArrayMsg[Msg,1]:=""
                ArrayMsg[Msg,5]:=0
                ArrayMsg[Msg,6]:=105
                ArrayMsg[Msg,7]:=0
                ArrayMsg[Msg,9]:=1

        CaseOf 3:
                ArrayMsg[Msg,1]:=""
                ArrayMsg[Msg,5]:=0
                ArrayMsg[Msg,6]:=105
                ArrayMsg[Msg,7]:=0
                ArrayMsg[Msg,9]:=3

        EndSwitch

        ControlText[1]:="" ControlText[2]:="" ControlText[3]:=""
        ControlText[4]:="" ControlText[5]:="" ControlText[6]:=""
        Buttons[1]:=0 Buttons[2]:=0 Buttons[3]:=0 Buttons[4]:=0
        Buttons[5]:=0 Buttons[6]:=0

        Switch(ArrayMsg[Msg,8])
        CaseOf 0:
```

```
CaseOf 1:
        ControlText[1]:= Bttn1          // "Continue"
        ControlText[2]:= Bttn2          // "Quit"
        Buttons[1]:=3
        Buttons[2]:=3

CaseOf 3:
        ControlText[1]:= Bttn0          // "OK"
        Buttons[1]:=3

CaseOf 4:
        ControlText[1]:= Bttn2          //  "Quit"
        Buttons[1]:=3

CaseOf 5:
        ControlText[1]:= Bttn6          // "Hint"
        ControlText[2]:= Bttn2          // "Quit"
        Buttons[1]:=3
        Buttons[2]:=3

CaseOf 6:
        ControlText[1]:= Bttn5          // "Yes"
        ControlText[2]:= Bttn4          //  "No"
        Buttons[1]:=3
        Buttons[2]:=3

CaseOf 8:
        ControlText[1]:= Bttn6          // "Hint"
        ControlText[2]:= Bttn1          // "Continue"
        ControlText[3]:= Bttn2          //  "Quit"
        Buttons[1]:=3
        Buttons[2]:=3
        Buttons[3]:=3
        ControlText[4]:= ArrayBttn[1,1]
        ControlText[5]:= ArrayBttn[1,2]
        ControlText[6]:= ArrayBttn[1,3]
        Buttons[4]:=1
        Buttons[5]:=1
        Buttons[6]:=1

EndSwitch
CoachMessageBox (
        0;                              // Name of the message box.
                                        // (Use in CoachMesssageBoxClose)
        ArrayMsg[Msg,1];                // The title to be displayed on the message box
        ArrayMsg[Msg,2];                // The text to be displayed on the message box
```

```
WPCHLink;                   // Handle of the coach bitmap dll
ArrayMsg[Msg,6];            // ID of the Coach Head Bitmap
ArrayMsg[Msg,7];            // ID of an additional bitmap to
                            // display on the message box
0;                          // The next three values specify
                            // the transparency color for the
                            // bitmap. The first value is the RED value
255;                        // The second value is the GREEN value
0;                          // The third value is the BLUE value
2;                          // Positon of head: In bubble = 2; Inside of Box = 1
ArrayMsg[Msg,5];            // Specifies the position of the bitmap
                            // on the dialog. Can be: Left!, Top!, Right!,Bottom!
ArrayMsg[Msg,4];            // Vertical Position of the message
                            // box (percentage of screen)
ArrayMsg[Msg,3];            // Horizontal Position of the message
                            // box (percentage of screen)
FontNameTitle;              // Font Name for the title
FontSizeTitle;              // Font Size for the title
FontName;                   // Font Name for the text
FontSize;                   // Font Size fot the text
ArrayMsg[Msg,9];            // Can be Simple!=1, Modal!=2, Modeless!=3
                            // (Simple uses the macro variable,
                            // Modal and Modeless use callbacks)
ChMsgBxCB@;                 // Callback label for modal or modeless
Result;                 // Macro Variable for simple dialog
ControlText[1];             // Name of a control to put on the
                            // message box (This name is passed to
                            // the callback function to identify
                            // the control user interacted with.
Buttons[1];                 // Type of control. Can be: Button!,
                            // CheckBox!, RadioButton!
ControlText[1];             // Text to put on the control
ControlText[2];             // The above three parameters can be
Buttons[2];                 // repeated for all controls desired.
ControlText[2];             // Text to put on the control
ControlText[3];             // The above three parameters can be
Buttons[3];                 // repeated for all controls desired.
ControlText[3];             // Text to put on the control
"RadBttn1";
Buttons[4];
ControlText[4];
"RadBttn2";
Buttons[5];
ControlText[5];
"RadBttn3";
Buttons[6];
```

```
                ControlText[6];
        )
RETURN

//*************************************************************
//      SUBROUTINE: ChMsgBxCB
//      DESC:
//*************************************************************
LABEL(ChMsgBxCB@)
        BttnFlg:=ChMsgBxCB[3]
RETURN



//*************************************************************
//      SUBROUTINE: DialogText
//      DESC:
//*************************************************************
LABEL(DialogText@)

DECLARE ArrayMsg[11,9]
DECLARE ArrayBttn[1,3]
DECLARE Buttons[6]
DECLARE ControlText[6]

// Message 1

ArrayMsg[1,1]:=     3
ArrayMsg[1,2]:=     "Choose from the following tasks, then choose \bOK\b when\n"+
                    "you are finished:\n"

ArrayBttn[1,1]:=    "Create your own prompts"
ArrayBttn[1,2]:=    "Paste the prompts you created"
ArrayBttn[1,3]:=    "Paste predefined personal prompts"

ArrayMsg[1,3]:=     85
ArrayMsg[1,4]:=     95
ArrayMsg[1,8]:=     8

// Message 2

ArrayMsg[2,1]:=     1
ArrayMsg[2,2]:=     "If you paste personal prompts into your template, the first time\n"+
                    "you use the template you may be prompted for personal information,\n"+
                    "but after that, the information will be filled in for you."
ArrayMsg[2,3]:=     15
```

```
ArrayMsg[2,4]:=      50
ArrayMsg[2,8]:=      3

// Message 3

ArrayMsg[3,1]:=      3
ArrayMsg[3,2]:=      "► Choose \bAdd\b"
ArrayMsg[3,3]:=      85
ArrayMsg[3,4]:=      95
ArrayMsg[3,8]:=      1

// Message 4

ArrayMsg[4,1]:=      3
ArrayMsg[4,2]:=      "► Type the prompt, then choose \bOK\b            "
ArrayMsg[4,3]:=      85
ArrayMsg[4,4]:=      95
ArrayMsg[4,8]:=      0

// Message 5

ArrayMsg[5,1]:=      3
ArrayMsg[5,2]:=      "► Choose \bOK\b.\n"
ArrayMsg[5,3]:=      85
ArrayMsg[5,4]:=      95
ArrayMsg[5,8]:=      0

// Message 6

ArrayMsg[6,1]:=      3
ArrayMsg[6,2]:=      "► Highlight a prompt in the list.\n"+
                     "► Click in the document window."
ArrayMsg[6,3]:=      85
ArrayMsg[6,4]:=      95
ArrayMsg[6,8]:=      1

// Message 7

ArrayMsg[7,1]:=      3
ArrayMsg[7,2]:=      "► Place the insertion point (cursor) where you\n"+
                     "   wish to paste the prompt.\n"+
                     "► Choose \bPaste\b."
ArrayMsg[7,3]:=      85
ArrayMsg[7,4]:=      95
ArrayMsg[7,8]:=      4
```

```
// Message 8

ArrayMsg[8,1]:=     1
ArrayMsg[8,2]:=     "You need to add prompts before you can paste them."
ArrayMsg[8,3]:=     50
ArrayMsg[8,4]:=     50
ArrayMsg[8,8]:=     3

// Message 9

ArrayMsg[9,1]:=     3
ArrayMsg[9,2]:=     "► Choose \bPersonal\b."

ArrayMsg[9,3]:=     85
ArrayMsg[9,4]:=     95
ArrayMsg[9,8]:=     1

// Message 10

ArrayMsg[10,1]:=    "\bQuit Coach?\b"

// Message 11

ArrayMsg[11,1]:=    3
ArrayMsg[11,2]:=    "► Highlight a prompt in the list.\n"+
                    "► Click in the document window.\n\n"+
                    "Choose \bClose\b when you are finished."
ArrayMsg[11,3]:=    85
ArrayMsg[11,4]:=    95
ArrayMsg[11,8]:=    4


Bttn0:="OK"                 // Assign Text for OK button.
Bttn1:="Continue"           // Assign Text for Continue button.
Bttn2:="Quit"               // Assign Text for Quit button.
Bttn3:="Show Me"            // Assign Text for Show Me button.
Bttn4:="No"                 // Assign Text for No button.
Bttn5:="Yes"                // Assign text for Yes button.
Bttn6:="Hint"               // Assign text for Hint button.

RETURN

Label(End@)
```