

```

*****
**      MACRO:      TCONVERT.WCM
**      PURPOSE:    Converts templates that were created for use with WordPerfect for
Windows 6.0a or 6.1.
*****
Application (A1; "WordPerfect"; Default; "EN")

MaxPrompts := 12                // Maximum number of prompts that fit in a dialog

Global PromptAbbrName := "Template Prompt Info"
Global BkmkAbbrName := "Template Bookmark Info"
Global OldPromptAbbrName := "Template Prompts"
Global OldBkmkAbbrName := "Template Bookmarks"
Global DelimiterChar := "|"

Global OldAddressPers := "<City, State Zip>"
Global NewAddress := "[City], [State] [ZIP Code]"
Global NewAddressPers := "<City>, <State> <ZIP Code>"
Global OldAddressPersField := 5
Global OldAddressField := 6
Global NumOldAddress := 0
Global OldAddress[MaxPrompts]

Global HasBookmarks := False
Global HasPrompts := False
Global HasOldBookmarks := False
Global HasOldPrompts := False

Global NumNewFld := 3           // Number of fields to replace
Global OldField[NumNewFld]     // Holds fields to replace
Global NewField[NumNewFld]     // Holds replacement fields
OldField[1] := "Account/ID"
NewField[1] := "Comments"
OldField[2] := "Telephone"
NewField[2] := "Phone Number"
OldField[3] := "Fax"
NewField[3] := "Fax Number"

// Set names of personal information fields
Declare DefFld[] := {"Name"; "Title"; "Organization"; "Address"; "City"; "State"; "ZIP Code";
"Phone Number"; "Fax Number"}
NumDefFld := Dimensions (DefFld[]; 1)
CloseSubdocs ()

SelectFile := False
Filename := ?Name

```

```

If (Filename = "")
    SelectFile := True
Else
    If (ToUpper (SubStr (Filename; StrLen (Filename)-3; 4)) <> "." + ToUpper (?
        DefaultTemplateExtension))
        SelectFile := True
    EndIf
EndIf
If (SelectFile)
    Filename := ""
    Label (Display@)
    DialogDefine ("Dlg1"; 50; 50; 164; 64; 1 + 2 + 16; "Convert Template")
    DialogAddFileNameBox ("Dlg1"; "Fb1"; 8; 8; 150; 14; 2; Filename; ?PathTemplate;
        "*.wpt")
    DialogAddText ("Dlg1"; "T1"; 8; 28; 150; 10; 1; "Select a template file to convert.")
    DialogDisplay ("Dlg1"; "Fb1")
    Button := MacroDialogResult
    DialogDestroy ("Dlg1")
    If (Button = 2)
        Quit
    EndIf
    Result := FindFile (Filename)
    If (Result = "")
        MessageBox (; "Template Not Found"; "The template file ^0 was not found,
            please include the full path and filename of the template."; Ok! | HasParameters!;
            Filename)
        Go (Display@)
    Else
        Filename := Result
    EndIf
    TemplateEdit (Filename)
    ExitWhenDone := True
Else
    ExitWhenDone := False
EndIf

If (?DocBlank)
    MessageBox (; "Template Is Blank"; "Template contains nothing to convert."; IconStop!)
    Quit
EndIf

If (GetAbbreviation (BkmkAbbrName) != "")
    HasBookmarks := True
EndIf
If (GetAbbreviation (PromptAbbrName) != "")
    HasPrompts := True

```

```

EndIf
OldTemplateBookmarks := GetAbbreviation (OldBkmkAbbrName)
If (OldTemplateBookmarks != "")
    HasOldBookmarks := True
EndIf
OldTemplatePrompts := GetAbbreviation (OldPromptAbbrName)
If (OldTemplatePrompts != "")
    HasOldPrompts := True
EndIf
If (HasBookmarks)
    JumpToAllDone := False
    If (HasOldBookmarks)
        AbbreviationDelete (OldBkmkAbbrName; CurrentDoc!)
        JumpToAllDone := True
    EndIf
    If (HasOldPrompts)
        AbbreviationDelete (OldPromptAbbrName; CurrentDoc!)
        JumpToAllDone := True
    EndIf
    If (JumpToAllDone)
        Go (AllDone@)
    Else
        MessageBox (; "Template Already Converted"; "This template has already been
converted."; IconInformation!)
        Quit
    EndIf
Else
    If ((Not HasOldBookmarks) And (Not HasOldPrompts))
        Go (CloseAndEnd@)
    EndIf
EndIf

WaitMsgInit ()
WaitMsgDisplay ("Converting template fields")

AddressChanges := CheckBkmkString (OldTemplateBookmarks; OldAddressPersField)
FieldCount := 0
If (Not AddressChanges)
    AddressChanges := CheckPromptString (OldTemplatePrompts; OldAddressField;
&FieldCount)
EndIf
If (FieldCount > MaxPrompts)
    MessageBox (; "Too Many Prompts"; "This template cannot be converted since the
conversion will create more than ^0 prompts."; IconExclamation! | HasParameters!;
MaxPrompts)
    Go (CloseAndEnd@)

```

```

EndIf

MatchSelection ()
SearchWrap (No!)
SearchFindWholeWordsOnly (No!)
SearchCaseSensitive (No!)
ConvertRenamedFields ()

TemplatePrompts := UpdatePromptsAbbrev (OldTemplatePrompts)

If (Not AddressChanges And Not HasBookmarks)
    // If there aren't any address changes, just update the abbreviations and rename the fields
    TemplateBookmarks := UpdateBookmarksAbbrev (OldTemplateBookmarks)
    Go (AllDone@)
EndIf

// There are address changes, so we need to update the document itself
ConvertAddressFields ()

PosDocVeryTop ()

// Delete the bookmarks from the document so that renumbering works correctly
GetData (NumBookmarks; Bookmark!; Count!; CurrentDoc!)
For (MarkNumber; 1; MarkNumber <= NumBookmarks; MarkNumber + 1)
    BookmarkDelete (MarkNumber)
EndFor

// Put Bookmarks around the Personal Information Fields
BkmrkString := ""
BkmrkCount := 1
For (Count; 1; Count <= NumDefFld; Count + 1)
    OnNotFound (NextDefFld@)
    SearchString ("<" + DefFld[Count] + ">")
    Repeat
        SearchNext (Extended!)
        BookmarkCreate (BkmrkCount)
        BkmrkString := BkmrkString + DefFld[Count] + DelimiterChar
        BkmrkCount := BkmrkCount + 1
    Until (False)

    Label (NextDefFld@)
    CloseSubdocs ()
    PosDocVeryTop ()
EndFor
OnNotFound ()

```

```

If (StrLen (BkmrkString) = 0)
    BkmrkString := BkmrkString + DelimiterChar
EndIf

// Put Bookmarks around the Template Prompts
If (TemplatePrompts = "")
    BkmrkString := BkmrkString + DelimiterChar
    Go (AlmostDone@)
EndIf
PullString := TemplatePrompts
ArrayCount := 0
While (PullString <> "")
    ArrayCount := ArrayCount + 1
    For (Loop; 0; Loop < 2; Loop + 1)
        NextDelimLoc := StrPos (PullString; DelimiterChar)
        If (NextDelimLoc = 0)
            AbbValue := PullString
            PullString := ""
            Go (LastOne@)
        EndIf
        PullStringLength := StrLen (PullString)
        AbbValue := SubStr (PullString; 1; NextDelimLoc-1)
        PullString := SubStr (PullString; NextDelimLoc + 1; PullStringLength)
        Label (LastOne@)
        If (Loop = 0)
            OnNotFound (NextAddFld@)
            SearchString ("[" + AbbValue + "]")
            Repeat
                SearchNext (Extended!)
                BookmarkCreate (BkmrkCount)
                BkmrkString := BkmrkString + DelimiterChar + ArrayCount
                BkmrkCount := BkmrkCount + 1
            Until (False)

            Label (NextAddFld@)
            CloseSubdocs ()
            PosDocVeryTop ()
        EndIf
    EndFor
EndWhile
OnNotFound ()

Label (AlmostDone@)
OnNotFound Call (ReadOnlyTemplate)
If (HasBookmarks)
    AbbreviationDelete (BkmkAbbrName; CurrentDoc!)

```

```

EndIf
If (HasOldBookmarks)
    AbbreviationDelete (OldBkmkAbbrName; CurrentDoc!)
EndIf
AbbreviationCreate (BkmkAbbrName; CurrentDoc!; BkmrkString)
OnNotFound ()

WaitMsgHide ()
WaitMsgDestroy ()

Label (AllDone@)
MessageBox (MsgBoxResult; "Conversion Complete"; "The template conversion is complete.
Do you wish to overwrite the original template?"; YesNoCancel! | DefButton2!)
OnCancel (End@)
Switch (MsgBoxResult)
    CaseOf 2:           // Cancel
        Go (End@)
    CaseOf 6:           // Yes
        FileSave (; WordPerfect_60!; Yes!)
    CaseOf 7:           // No
        FileSaveAsDlg ()
EndSwitch

Label (CloseAndEnd@)
If (ExitWhenDone)
    Close ()
EndIf

Label (End@)
Quit

/*****
/**  PROCEDURE: ReadOnlyTemplate
/**  INPUT VARIABLES: None
/**  OUTPUT VARIABLES: None
/**  DESCRIPTION: Specifies when a 'not found' has occurred during the abbreviation
tokens - probably due to a read-only template.
*****/
Procedure ReadOnlyTemplate ()
WaitMsgHide ()
WaitMsgDestroy ()
MessageBox (; "Read Only Template"; "An unexpected 'Not Found' condition occurred. The
template file may be marked read only. You may be able to make a writable copy of the template
and play TCONVERT.WCM again."; IconStop!)
Quit
EndProc

```

```

//*****
//*****
//*   PROCEDURE: CloseSubdocs
//*   INPUT: None
//*   OUTPUT: None
//*   DESCRIPTION: Exits all Sub Documents (headers, text boxes, etc.).
//*****
Procedure CloseSubdocs ()
While (?Substructure)
    SubDoc := ?CurrentSubDoc
    SubstructureExit ()
    If ((SubDoc = 10) Or (SubDoc = 11))
        BoxEnd ()
    EndIf
EndWhile
EndProc
//*****

//*****
//   FUNCTION NAME: GetAbbreviation
//   IN: Name of abbreviation to retrieve
//   RETURNS: Contents of abbreviation or NULL string
//   USES: Nothing
//   DESCRIPTION: Retrieves the contents of a specified abbreviation.
//*****
Function GetAbbreviation (AbbrName)
SingleString := ""
GetData (NumOfAbbrs; Abbreviation!; Count!; CurrentDoc!)
For (CurrAbbrNum; 1; CurrAbbrNum <= NumOfAbbrs; CurrAbbrNum + 1)
    GetData (CurrAbbrName; Abbreviation!; Name!; CurrentDoc!; CurrAbbrNum)
    If (CurrAbbrName = AbbrName)
        GetData (SingleString; Abbreviation!; Data!; CurrentDoc!; CurrAbbrNum)
        Break
    EndIf
EndFor
Return (SingleString)
EndFunc
//*****

//*****
//   FUNCTION NAME: FindFile
//   IN: Name of file to search for
//   RETURNS: Path of file or NULL string
//   USES: Nothing
//   DESCRIPTION: Locates a file in the template directories and the current directory.

```

```

//*****
Function FindFile (Name)
If (Name = "")
    Return (")
EndIf
NewName := ?PathTemplate + Name
FileExists (FExists; NewName)
If (FExists)
    Return (NewName)
Else
    NewName := ?PathTemplateSupplemental + Name
EndIf
FileExists (FExists; NewName)
If (FExists)
    Return (NewName)
EndIf
FileExists (FExists; Name)
If (FExists)
    Return (Name)
EndIf
Return (")
EndFunc
//*****

```

```

//*****
//    FUNCTION NAME: CheckBkmkString
//    IN: InString; CheckString
//    RETURNS: True or False
//    USES: Nothing
//    DESCRIPTION: Checks if a specified field is in a composite string.
//*****
Function CheckBkmkString (InString; CheckString)
StartLoc := 1
DelimLoc := 1
Len := StrLen (InString)
Repeat
    DelimLoc := StrPos (SubStr (InString; StartLoc; Len-StartLoc + 1); DelimiterChar)
    Field := SubStr (InString; StartLoc; DelimLoc - 1)
    If (Field = CheckString)
        Return (True)
    EndIf
    If (Field = "")
        Return (False)
    EndIf
    StartLoc := StartLoc + DelimLoc
Until (DelimLoc = 0)

```


Return (False)

EndFunc

// FUNCTION NAME: CheckPromptString

// IN: InString; CheckString

// RETURNS: True or False

// USES: Nothing

// DESCRIPTION: Checks if a specified field is in a composite string.

Function CheckPromptString (InString; CheckString; &Counter)

StartLoc := 1

DelimLoc := 1

RetVal := False

Len := StrLen (InString)

Repeat

 DelimLoc := StrPos (SubStr (InString; StartLoc; Len-StartLoc + 1); DelimiterChar)

 Field := SubStr (InString; StartLoc; DelimLoc - 1)

 StartLoc := StartLoc + DelimLoc

 DelimLoc := StrPos (SubStr (InString; StartLoc; Len-StartLoc + 1); DelimiterChar)

 If (DelimLoc = 0)

 Link := SubStr (InString; StartLoc; Len)

 Else

 Link := SubStr (InString; StartLoc; DelimLoc - 1)

 EndIf

 StartLoc := StartLoc + DelimLoc

 Counter := Counter + 1

 If (Link = CheckString)

 RetVal := True

 Counter := Counter + 2

 EndIf

Until (DelimLoc = 0)

Return (RetVal)

EndFunc

// FUNCTION NAME: UpdatePromptsAbbrev

// IN: OldTemplatePrompts

// RETURNS: New Template Prompts Abbreviation

// USES: DelimiterChar; OldAddressField

// DESCRIPTION: Updates the Template Prompts abbreviation.

Function UpdatePromptsAbbrev (OldTemplatePrompts)

If (OldTemplatePrompts = "")

```

Return ("" )
EndIf
NewABField[] := {"0"; "Name"; "Greeting"; "Title"; "Organization"; "Address"; "City|City|
State|State|Zip Code|Zip Code"; "Phone Number"; "Fax Number"; "Comments"}
NewAbbrev := ""
StartLoc := 1
DelimLoc := 1
Len := StrLen (OldTemplatePrompts)
Repeat
    DelimLoc := StrPos (SubStr (OldTemplatePrompts; StartLoc; Len-StartLoc + 1);
    DelimiterChar)
    Field := SubStr (OldTemplatePrompts; StartLoc; DelimLoc - 1)
    StartLoc := StartLoc + DelimLoc
    DelimLoc := StrPos (SubStr (OldTemplatePrompts; StartLoc; Len-StartLoc + 1);
    DelimiterChar)
    If (DelimLoc = 0)
        Link := SubStr (OldTemplatePrompts; StartLoc; Len)
    Else
        Link := SubStr (OldTemplatePrompts; StartLoc; DelimLoc - 1)
    EndIf
    StartLoc := StartLoc + DelimLoc
    If (Link != OldAddressField)
        NewAbbrev := NewAbbrev + Field + DelimiterChar + NewABField[Link + 1] +
        DelimiterChar
    Else
        NumOldAddress := NumOldAddress + 1
        OldAddress[NumOldAddress] := Field
        NewAbbrev := NewAbbrev + NewABField[Link + 1] + DelimiterChar
    EndIf
Until (DelimLoc = 0)
// Get rid of the extra pipe
If (SubStr (NewAbbrev; StrLen (NewAbbrev); 1) = DelimiterChar)
    NewAbbrev := SubStr (NewAbbrev; 1; StrLen (NewAbbrev)-1)
EndIf
OnNotFound Call (ReadOnlyTemplate)
If (HasOldPrompts)
    AbbreviationDelete (OldPromptAbbrName; CurrentDoc!)
EndIf
If (HasPrompts)
    AbbreviationDelete (PromptAbbrName; CurrentDoc!)
EndIf
AbbreviationCreate (PromptAbbrName; CurrentDoc!; NewAbbrev)
OnNotFound ()
Return (NewAbbrev)
EndFunc
//*****

```

```

//*****
//      FUNCTION NAME: UpdateBookmarksAbbrev
//      IN: OldTemplateBookmarks
//      RETURNS: New Template Bookmarks Abbreviation
//      USES: DelimiterChar; OldAddressPersField
//      DESCRIPTION: Updates the Template Bookmarks abbreviation.
//*****
Function UpdateBookmarksAbbrev (OldTemplateBookmarks)
NewABField[] := {"Name"; "Title"; "Organization"; "Address"; "City|State|Zip Code"; "Phone
Number"; "Fax Number"}
NewAbbrev := ""
StartLoc := 1
DelimLoc := 1
If (StrLeft (OldTemplateBookmarks; 2) = (DelimiterChar + DelimiterChar))
    NewAbbrev := OldTemplateBookmarks
Else
    Len := StrLen (OldTemplateBookmarks)
    Repeat
        DelimLoc := StrPos (SubStr (OldTemplateBookmarks; StartLoc; Len-StartLoc +
1); DelimiterChar)
        Field := SubStr (OldTemplateBookmarks; StartLoc; DelimLoc - 1)
        StartLoc := StartLoc + DelimLoc
        If (Field != "")
            NewAbbrev := NewAbbrev + NewABField[Field] + DelimiterChar
        Else
            DelimLoc := StrPos (SubStr (OldTemplateBookmarks; StartLoc; Len-
StartLoc + 1); DelimiterChar)
            If (DelimLoc = 0)
                NewAbbrev := NewAbbrev + DelimiterChar
            Else
                NewAbbrev := NewAbbrev + SubStr (OldTemplateBookmarks;
DelimLoc + 1; Len)
                DelimLoc := 0
            EndIf
        EndIf
    Until (DelimLoc = 0)
EndIf
OnNotFound Call (ReadOnlyTemplate)
If (HasOldBookmarks)
    AbbreviationDelete (OldBkmkAbbrName; CurrentDoc!)
EndIf
If (HasBookmarks)
    AbbreviationDelete (BkmkAbbrName; CurrentDoc!)
EndIf
AbbreviationCreate (BkmkAbbrName; CurrentDoc!; NewAbbrev)

```

```

OnNotFound ()
Return (NewAbbrev)
EndFunc
//*****

//*****
//      PROCEDURE NAME: ConvertAddressFields
//      IN: None
//      RETURNS: Nothing
//      USES: OldAddress; NewAddressPers; NewAddress
//      DESCRIPTION: Updates the Address fields which have changed format.
//*****

Procedure ConvertAddressFields ()
PosDocVeryTop ()
OnNotFound (NextOld@)
SearchString (OldAddressPers)
ReplaceString (NewAddressPers)
ReplaceAll (Extended!)
Label (NextOld@)
OnNotFound (LastOld@)
CloseSubDocs ()
PosDocVeryTop ()
For (x; 1; x <= NumOldAddress; x + 1)
    SearchString ("[" + OldAddress[x] + "]")
    ReplaceString (NewAddress)
    ReplaceAll (Extended!)
EndFor
Label (LastOld@)
OnNotFound ()
CloseSubDocs ()
EndProc
//*****

//*****
//      PROCEDURE NAME: ConvertRenamedFields
//      IN: None
//      RETURNS: Nothing
//      USES:
//      DESCRIPTION: Updates the Address fields which have changed name.
//*****

Procedure ConvertRenamedFields ()
For (x; 1; x <= NumNewFld; x + 1)
    OnNotFound (NextOld@)
    SearchString ("<" + OldField[x] + ">")
    ReplaceString ("<" + NewField[x] + ">")
    ReplaceAll (Extended!)

```

```

Label (NextOld@)
OnNotFound (LastOld@)
CloseSubDocs ()
PosDocVeryTop ()
SearchString ("[" + OldField[x] + "]")
ReplaceString ("[" + NewField[x] + "]")
ReplaceAll (Extended!)
Label (LastOld@)
CloseSubDocs ()
PosDocVeryTop ()

EndFor
OnNotFound ()
EndProc
//*****

//*****
//      PROCEDURE: WaitMsgInit ()
//      PURPOSE: Initializes macro wait messages.
//*****
PROCEDURE WaitMsgInit ()
DialogDefine ("WaitMsg"; 50; 50; 150; 36; 16; "Please Wait")
DialogAddText ("WaitMsg"; "WaitText"; 8; 14; 144; 16; 1; "")
DialogLoad ("WaitMsg")
Return
ENDPROC

//*****
//      PROCEDURE: WaitMsgDisplay (Text)
//      PURPOSE: Displays a previously defined wait message dialog.
//*****
PROCEDURE WaitMsgDisplay (Text)
RegionSetWindowText ("WaitMsg"+ ".WaitText"; Text)
DialogShow ("WaitMsg";; WaitDlgCallBack)
ENDPROC

PROCEDURE WaitDlgCallBack ()
If (WaitDlgCallBack[5] = 274 And WaitDlgCallBack[6] = 61536)
    Assert (CancelCondition!)
EndIf
ENDPROC

//*****
//      PROCEDURE: WaitMsgHide
//      PURPOSE: Hides a previously defined wait message dialog.
//*****
PROCEDURE WaitMsgHide ()

```

