```
//*************************************************************
//*       MACRO: PARABRK.WCM
//*       PURPOSE: Puts a choice of paragraph breaks at the end of the paragraph in which the insertion point is located. If you position the insertion point at a blank line the break will be positioned on that line.
//*       NOTE: Either the Default or the Supplemental macro folder must be set to the WordPerfect install's Macros\WPWin directory.
//*************************************************************
Application (A1; "WordPerfect"; Default; "EN")

EntryCount := 20       // This must match the number of paragraph breaks defined
Spacing := 3
RptVal := 3
BreakSize := ?FontSize
WPGDir := ?PathMacros ()
If (WPGDir = "")
        WPGDir := ?PathGraphicsSupplemental
EndIf
If (WPGDir = "")
        WPGDir := ?PathProgram
        Call (CheckDir@)
EndIf
If (SubStr (WPGDir; StrLen (WPGDir); 1) <>"\")
        WPGDir := WPGDir + "\"
EndIf

If (?SubStructure)
        MessageBox (; "Error - Not In Main Edit Screen"; "This macro can only be run from the main editing screen."; IconStop!)
        Go (End@)
EndIf

MacroStatusPrompt (On!; "Initializing...")

Declare (ParaBrk[EntryCount + 1; 4])
                // Array is set up as follows: [ NUMBER; Element ] where NUMBER is the maximum number of paragraph break types and ELEMENT is the number of parts for each type. The Elements are:
                //        1 - Name of the paragraph break (WPstring)
                //        2 - Character for the break (WPstring)
                //                For types 1 and 2 (characters), this is the actual characters for the break. For type 3 (graphic), this is the filename for the .wpg file.
                //        3 - Type of character
                //                This is either WPCharacters (1), Wingdings (2), Square Graphic (3) or Long Graphic (4).
                //        4 - LastChar
```

```
                //              For the character types, this is the actual character for the end of
                the break.
                //              For the graphic type, this is the filename for the end of the break. If
                this filename is the same as the first, the image will be flipped on the y axis
                (mirror image). This is only valid for Square Graphics.
                //              IMPORTANT - If you do not want your character to act like it is
                "paired" (there is a left and a right), then DO NOT assign this element in the array.
ParaBrk[1; 1] := "Dots"
ParaBrk[1; 2] := "○" // 4,1
ParaBrk[1; 3] := 1

ParaBrk[2; 1] := "Squares"
ParaBrk[2; 2] := "■" // 4,2
ParaBrk[2; 3] := 1

ParaBrk[3; 1] := "Diamonds"
ParaBrk[3; 2] := "♢" // 5,1
ParaBrk[3; 3] := 1

ParaBrk[4; 1] := "Triangles"
ParaBrk[4; 2] := "▼" // 6,30
ParaBrk[4; 3] := 1

ParaBrk[5; 1] := "Stars"
ParaBrk[5; 2] := "⋆" // 6,112
ParaBrk[5; 3] := 1

ParaBrk[6; 1] := "Pound Signs"
ParaBrk[6; 2] := "#"
ParaBrk[6; 3] := 1

ParaBrk[7; 1] := "Flowers"
ParaBrk[7; 2] := "{"
ParaBrk[7; 3] := 2

ParaBrk[8; 1] := "Floral"
ParaBrk[8; 2] := "𒀭"   // 12,150
ParaBrk[8; 3] := 2
ParaBrk[8; 4] := "𒀱"   // 12,151

ParaBrk[9; 1] := "Fancy 1"
ParaBrk[9; 2] := "𒁍"   // 12,154
ParaBrk[9; 3] := 2
ParaBrk[9; 4] := "𒁕"   // 12,155

ParaBrk[10; 1] := "Fancy 2"
```

```
ParaBrk[10; 2] := "¦" // 12,207
ParaBrk[10; 3] := 2
ParaBrk[10; 4] := " "  // 12,210

ParaBrk[11; 1] := "Fleur-de-lis (wpg)"
ParaBrk[11; 2] := "Ender01.wpg"
ParaBrk[11; 3] := 3

ParaBrk[12; 1] := "Star (wpg)"
ParaBrk[12; 2] := "Ender02.wpg"
ParaBrk[12; 3] := 3

ParaBrk[13; 1] := "Brush Stroke Star (wpg)"
ParaBrk[13; 2] := "Ender03.wpg"
ParaBrk[13; 3] := 3

ParaBrk[14; 1] := "Square Floral (wpg)"
ParaBrk[14; 2] := "Ender04.wpg"
ParaBrk[14; 3] := 3

ParaBrk[15; 1] := "Fancy Floral (wpg)"
ParaBrk[15; 2] := "Ender05.wpg"
ParaBrk[15; 3] := 3
ParaBrk[15; 4] := "Ender05.wpg"

ParaBrk[16; 1] := "Snowflake (wpg)"
ParaBrk[16; 2] := "Ender06.wpg"
ParaBrk[16; 3] := 3

ParaBrk[17; 1] := "Line 1 (wpg)"
ParaBrk[17; 2] := "Ender07.wpg"
ParaBrk[17; 3] := 4

ParaBrk[18; 1] := "Line 2 (wpg)"
ParaBrk[18; 2] := "Ender08.wpg"
ParaBrk[18; 3] := 4

ParaBrk[19; 1] := "Line 3 (wpg)"
ParaBrk[19; 2] := "Ender09.wpg"
ParaBrk[19; 3] := 4

ParaBrk[20; 1] := "Line 4 (wpg)"
ParaBrk[20; 2] := "Ender10.wpg"
ParaBrk[20; 3] := 4

If (?DocBlank)
```

```
        Go (MainMenu@)
EndIf
If ((?RightCode = 204) And (?LeftCode = 204))
        Go (MainMenu@)
EndIf
PosParagraphNext ()
If ((?RightCode = 0) And (?RightChar = ""))
        If (?LeftCode <>204)
                HardReturn ()
        EndIf
        Go (MainMenu@)
EndIf
PosLineUp ()
PosLineEnd ()
If (?LeftCode = 204 )
        Go (MainMenu@)
EndIf
HardReturn ()
Label (MainMenu@)
MacroStatusPrompt (On!; "Initializing dialog...")
DialogDefine ("Dlg1"; 50; 50; 199; 84; 16; "Paragraph Breaks")
DialogAddListBox ("Dlg1"; "Lb1"; 8; 8; 125; 72; 1; Choice)
For (Counter; 1; Counter <= EntryCount; Counter + 1)
        DialogAddListItem ("Dlg1"; "Lb1"; ParaBrk[Counter; 1])
EndFor
DialogAddPushButton ("Dlg1"; "OKBttn"; 141; 8; 50; 14; DefaultBttn! | OKBttn!; "OK")
DialogAddPushButton ("Dlg1"; "CancelBttn"; 141; 26; 50; 14; CancelBttn!; "Cancel")
DialogAddPushButton ("Dlg1"; "Pb3"; 141; 44; 50; 14; 0; "C&ustom")
DialogLoad ("Dlg1")
hwndLb1 := DialogHandle ("Dlg1"; "Lb1")
UserLink := DLLLoad ("User32")
        // Select the first item
DLLCall (UserLink; "SendMessageA"; nVoid:INTEGER; {hwndLb1; 0186x; 0; 0})
DllFree (UserLink)
Label (DisplayDialog@)
MacroStatusPrompt (Off!)
Display (On!)
DialogShow ("Dlg1"; "Lb1")
Result := MacroDialogResult
Display (Off!)
If (Result = 2)
        DialogDestroy ("Dlg1")
        Go (End@)
EndIf
If (Result = "Pb3")
        DialogDefine ("Dlg2"; 50; 50; 153; 80; 1 + 2 +  16; "Custom Paragraph Break")
```

```
        DialogAddText ("Dlg2"; "T1"; 8; 10; 95; 10; 1; "&Character to use:")
        DialogAddEditBox ("Dlg2"; "E1"; 112; 8; 31; 14; WPChars!; UserChar; 10)
        DialogAddText ("Dlg2"; "T2"; 8; 27; 95; 10; 1; "&Number of characters:")
        DialogAddCounter ("Dlg2"; "C1"; 111; 26; 35; 14; 0; RptVal; 1; 99; 1)
        DialogAddText ("Dlg2"; "T3"; 8; 45; 95; 10; 1; "&Spaces between characters:")
        DialogAddCounter ("Dlg2"; "C2"; 111; 43; 35; 14; 0; Spacing; 0; 99; 1)
        Display (On!)
        DialogShow ("Dlg2"; "E1")
        Result2 := MacroDialogResult
        DialogDestroy ("Dlg2")
        Display (Off!)
        If (Result2 = 1)
                Choice := EntryCount + 1
                ParaBrk[Choice; 1] := "User"
                ParaBrk[Choice; 2] := UserChar
                ParaBrk[Choice; 3] := 1
                Go (Spacing@)
        EndIf
        Go (DisplayDialog@)
EndIf
For (Counter; 1; ParaBrk[Counter; 1] <>Choice; Counter + 1)
EndFor
Label (Spacing@)
If (ParaBrk[Counter; 3] = 3 Or ParaBrk[Counter; 3] = 4)
        FileExists (FileExist; WPGDir + ParaBrk[Counter; 2])
        If (Not FileExist)
                MessageBox (MsgBoxResult; "Graphic Not Found"; "^0 does not exist.  Check
                your graphics directory in preferences, or press Retry to select a different
                directory."; IconExclamation! | RetryCancel! | HasParameters! | Beep!; WPGDir +
                ParaBrk[Counter; 2])
                If (MsgBoxResult = 4)
                        Call (CheckDir@)
                EndIf
                Go (DisplayDialog@)
        EndIf
EndIf
MacroStatusPrompt (On!; "Placing paragraph break...")
For (Spaces; ""; StrLen (Spaces) < Spacing; Spaces + " ")
EndFor
If (ParaBrk[Counter; 3] < 3)
        CompoundString := ""
        If (Exists (ParaBrk[Counter; 4]) <> 0)
                CompoundString := ParaBrk[Counter; 2] + Spaces + ParaBrk[Counter; 4]
        Else
                For (LoopCount; 0; LoopCount < RptVal - 1; LoopCount + 1)
                        CompoundString := CompoundString + ParaBrk[Counter; 2] + Spaces
```

```
                EndFor
                CompoundString := CompoundString + ParaBrk[Counter; 2]
        EndIf
EndIf
DialogDestroy ("Dlg1")
Case Call (ParaBrk[Counter; 3]; {
        1; WPChars@;
        2; Wingdings@;
        3; SquareGraphics@;
        4; LongGraphics@
        }; End@)

If (ParaBrk[Counter; 3] <>4)
        PosLineVeryEnd ()
        If (?RightCode = 0 And ?RightChar = "")
                HardReturn ()
        EndIf
EndIf
PosParagraphNext ()

Label (End@)
Quit

//*****************************************************************
//*      ROUTINE: WPChars@
//*      INPUT VARIABLES: CompoundString
//*      OUTPUT VARIABLES: none
//*      DESCRIPTION: Places CompoundString in document.
//*****************************************************************
Label (WPChars@)
        Center ()
        Type (CompoundString)
        EndCenterOrAlignment ()
Return
//*****************************************************************

//*****************************************************************
//*      ROUTINE: Wingdings@
//*      INPUT VARIABLES: CompoundString
//*      OUTPUT VARIABLES: none
//*      DESCRIPTION: Places CompoundString in document, then changes font to Wingdings.
//*****************************************************************
Label (Wingdings@)
        Center ()
        Type (CompoundString)
        EndCenterOrAlignment ()
```

```
        SelectLineBegin ()
        Font ("Wingdings")
        SelectMode (Off!)
Return
//*****************************************************************

//*****************************************************************
//*      ROUTINE: SquareGraphics@
//*      INPUT VARIABLES: Spaces; RptVal
//*      OUTPUT VARIABLES: none
//*      DESCRIPTION: Places a multiple number of small square graphics in document.
//*****************************************************************
Label (SquareGraphics@)
        FlipX := False
        Center ()
        If (Exists (ParaBrk[Counter; 4]) = 0)
                For (LoopCount; 0; LoopCount < RptVal - 1; LoopCount + 1)
                        Call (PlaceGraphics@)
                        Type (Spaces)
                EndFor
        Else
                Call (PlaceGraphics@)
                If (ParaBrk[Counter; 2] = ParaBrk[Counter; 4])
                        FlipX := True
                EndIf
                Type (Spaces)
        EndIf
        Call (PlaceGraphics@)
        EndCenterOrAlignment ()
Return
//*****************************************************************

//*****************************************************************
//*      ROUTINE: LongGraphics@
//*      INPUT VARIABLES: BreakSize
//*      OUTPUT VARIABLES: none
//*      DESCRIPTION: Places a single graphic in center of page or column.
//*****************************************************************
Label (LongGraphics@)
        If (?Column >0)
                GraphicWidth := ?ColumnWidth/2
        Else
                GraphicWidth := (?PaperWidth-MarginLeft()-MarginRight())/2
        EndIf
        BoxCreate (UserBox!)
        BoxAttachTo (Paragraph!)
```

```
        BoxHorizontalAlignment (AlignMargins!; Center!; 0)
        //BoxVerticalAlignment (Top!; 0)      // ab: this doesn't work, replaced with the following
        BoxVerticalAlignment (Baseline!)
        BoxContentPreserveAspectRatio (State: Yes!)
        //BoxHeight (AutoHeight!)    // ab: this doesn't work, replaced with the following
        BoxHeight (BreakSize*1w)
        BoxWidth (GraphicWidth)
        BorderSetSpacing (Yes!)
        BorderInsideSpacing (0.0"; 0.0"; 0.0"; 0.0")
        BorderOutsideSpacing (0.0"; 0.0"; 0.0"; 0.0")
        BoxTextFlow (NeitherSide!)
        BoxContentType (Image!)
        BoxImageRetrieve (MakeInternal!; WPGDir + ParaBrk[Counter; 2])
        BoxEnd (Save!)
        CloseGraphicsControlBar ()
Return
//****************************************************************

//****************************************************************
//*      ROUTINE: PlaceGraphics@
//*      INPUT VARIABLES: BreakSize
//*      OUTPUT VARIABLES: none
//*      DESCRIPTION: Places a single small graphic in document.
//****************************************************************
Label (PlaceGraphics@)
        BookmarkCreate ("Paragraph Break Macro")
        BoxCreate (UserBox!)
        BoxAttachTo (Character!)
        BoxVerticalAlignment (Baseline!)
        BoxContentPreserveAspectRatio (State: Yes!)
        BoxHeight (BreakSize*1w)
        //BoxWidth (AutoWidth!)      // ab: this doesn't work, replaced with the following
        BoxWidth (BreakSize*1w)
        BorderSetSpacing (Yes!)
        BorderInsideSpacing (0.0"; 0.0"; 0.0"; 0.0")
        BorderOutsideSpacing (0.0"; 0.0"; 0.0"; 0.0")
        BoxContentType (Image!)
        BoxImageRetrieve (MakeInternal!; WPGDir + ParaBrk[Counter; 2])
        BoxContentPreserveAspectRatio (State: Yes!)
        If (FlipX)
                BoxImageFlipX (Yes!)
        EndIf
        BoxEnd (Save!)
        CloseGraphicsControlBar ()
        SelectCharPrevious ()
        EditCut ()
```

```
        BookmarkFind ("Paragraph Break Macro")
        EditPaste ()
        BookmarkDelete ("Paragraph Break Macro")
Return
//*****************************************************************

//*****************************************************************
//*      ROUTINE: CheckDir@
//*      INPUT VARIABLES: WPGDir
//*      OUTPUT VARIABLES: WPGDir
//*      DESCRIPTION: This will create a dialog to prompt for a directory.
//*****************************************************************
Label (CheckDir@)
DialogDefine ("PathDlg"; 50; 50; 172; 90; 1 + 2 + 16; "Graphic Directory")
DialogAddFilenameBox ("PathDlg"; "Fb1"; 8; 8; 150; 14; 1; WPGDir; WPGDir; "*.wpg")
DialogAddText ("PathDlg"; "T1"; 8; 28; 150; 20; 1; "Please select the directory that contains the
Paragraph Break graphics files.")
Label (Redisplay@)
Display (On!)
DialogShow ("PathDlg"; "Fb1")
Display (Off!)
If (MacroDialogResult = 2)
        DialogDestroy ("PathDlg")
        Go (End@)
EndIf
If (WPGDir = "")
        Go (Redisplay@)
EndIf
DialogDestroy ("PathDlg")
If (SubStr (WPGDir; StrLen (WPGDir); 1) <>"\")
        WPGDir := WPGDir + "\"
EndIf
Return
//*****************************************************************
```