```
//*************************************************************
//*       MACRO: REVERSE.WCM
//*       PURPOSE: Applies colors and fill style to selected text or table cells.
//*************************************************************
Application (A1; "WordPerfect"; Default; "EN")

DefinedColors := 16
DefinedPatterns := 12
WaitMsgInit ()
WaitMsgDisplay ("Initializing")
Call (Declarations@)
                                    // Set defaults here...
TxtColor := Clr[1; 1]               // white
FillColor := Clr[2; 1]              // black
PatFill := Pat[2; 1]                // 100% Fill
CntrTxt := 0
LockCell := 0
HeaderRow := 0
Paragraph := 0
TextBox := 1
EndOfDoc := False
CodesOn := False

If (?RevealCodesActive)
        RevealCodes (Off!)
        CodesOn := True
EndIf

If (?TableInTable = 0)
        InTable := False
Else
        InTable := True
        InParagraph := False
        InTextBox := False
EndIf

If (?BlockActive = 0)
        If (InTable)
                SelectCell ()
        Else
                SelectWord ()
        EndIf
        If (?BlockActive = 0) // Double check to see if something was selected
                MessageBox (; "No Items Selected"; "You must select the text or table cells you
                would like to modify before running this macro."; IconStop!)
                Go (End@)
```

```
        EndIf
EndIf

StatusPrompt ("Initializing dialog...")

If (InTable)
        DlgHeight := 125
Else
        DlgHeight := 137
EndIf
DialogDefine ("Dlg1"; 50; 50; 204; DlgHeight; 1 + 2 + 16; "Reverse Text Options")
DialogAddGroupBox ("Dlg1"; "Gb1"; 8; 5; 90; 55; "&Text color")
DialogAddComboBox ("Dlg1"; "Cmb1"; 18; 18; 70; 50; 16; TxtColor; 20)

DialogAddGroupBox ("Dlg1"; "Gb2"; 106; 5; 90; 55; "&Fill style/color")
DialogAddComboBox ("Dlg1"; "Cmb2"; 116; 18; 70; 50; 16; PatFill; 20)
For (Count; 1; Count <= DefinedPatterns; Count + 1)
        DialogAddListItem ("Dlg1"; "Cmb2"; Pat[Count; 1])
EndFor

DialogAddComboBox ("Dlg1"; "Cmb3"; 116; 36; 70; 50; 16; FillColor; 20)

For (Count; 1; Count <= DefinedColors; Count + 1)
        DialogAddListItem ("Dlg1"; "Cmb1"; Clr[Count; 1])
        DialogAddListItem ("Dlg1"; "Cmb3"; Clr[Count; 1])
EndFor

If (InTable)
        DialogAddGroupBox ("Dlg1"; "Gb3"; 8; 65; 188; 34; "Table &options")
        DialogAddCheckBox ("Dlg1"; "Cb1"; 18; 78; 50; 11; "C&enter text"; CntrTxt)
        DialogAddCheckBox ("Dlg1"; "Cb2"; 76; 78; 50; 11; "&Lock cell"; LockCell)
        DialogAddCheckBox ("Dlg1"; "Cb3"; 134; 78; 52; 11; "&Header row"; HeaderRow)
Else
        DialogAddGroupBox ("Dlg1"; "Gb3"; 8; 65; 188; 46; "Text &options")
        DialogAddRadioButton ("Dlg1"; "Cb1"; 18; 78; 127; 11; "Place selected text in text
        &box"; TextBox)
        DialogAddRadioButton ("Dlg1"; "Cb2"; 18; 90; 127; 11; "Apply attributes to whole
        &paragraph"; Paragraph)
EndIf
StatusPrompt ("Off")
Display (On!)
WaitMsgHide ()
DialogDisplay ("Dlg1"; "Cmb1")
Button := MacroDialogResult
DialogDestroy ("Dlg1")
Display (Off!)
```

```
If (Not InTable)
	If (Paragraph = 1)
		InParagraph := True
		InTextBox := False
	Else
		InParagraph := False
		InTextBox := True
	EndIf
EndIf
If (Button = 2)
	Go (End@)
Else
If (Button = 1)
	WaitMsgDisplay ("Applying attributes")
		//	FIRST DO THE SETUP
	If (InTextBox)
		StatusPrompt ("Creating in line text box...")
		SpaceCleanup := ?CleanupSpaces
		CleanUpSpaces (Off!)
		EditCut ()
		CleanUpSpaces (SpaceCleanup)
		BookmarkCreate ("Heading Macro Place Holder")
		BoxCreate (TextBox!)
		TextBoxNumber := ?BoxNumber
		BoxAttachTo (Character!)
		BoxVerticalAlignment (Baseline!)
		BoxContentEdit ()
		EditPaste ()
		PosDocTop ()
	Else
	If (InParagraph)
		SelectParagraph ()
		PosSelectBottom ()
		If (?RightCode = 0 And ?RightChar = "")
			EndOfDoc := True
			SelectOff ()
			HardReturn ()
			SelectParagraphPrevious ()
		EndIf
	EndIf EndIf
		//	NOW SET THE COLOR FOR THE TEXT
	If (Not InTextBox)
		StatusPrompt ("Setting text color...")
	EndIf
	For (SelClr; 1; Clr[SelClr; 1] <>TxtColor; SelClr + 1)
	EndFor
```

```
        TextColor (; Clr[SelClr; 2]; Clr[SelClr; 3]; Clr[SelClr; 4])
        If (InTextBox)
                SubstructureExit ()
                BoxWidth (AutoWidth!)
                BoxBorder (0)
        EndIf
                //      NOW SET THE FILL STYLE
        If (Not InTextBox)
                StatusPrompt ("Setting fill style...")
        EndIf
        For (SelPat; 1; Pat[SelPat; 1] <>PatFill; SelPat + 1)
        EndFor
        If (InTable)
                TableCellFillStyle (Pat[SelPat; 2])
        Else
        If (InParagraph)
                TextBorderCreate (SpacingOnly!; Pat[SelPat; 2]; ParagraphBorder!)
        Else
        If (InTextBox)
                BorderSetSpacing (Yes!)
                BorderOutsideSpacing (0; 0; 0; 0)
                BoxFill (Pat[SelPat; 2])
        EndIf EndIf EndIf
                //      NOW SET THE FILL COLORS
        If (Not InTextBox)
                StatusPrompt ("Setting fill colors...")
        EndIf
        For (SelFClr; 1; Clr[SelFClr; 1] <>FillColor; SelFClr + 1)
        EndFor
        If (Pat[SelPat; 2] < 10)
                FillPercent := (Pat[SelPat; 2] + 1)*10
                If (InTable)
                        TableCellFillColors (; Clr[SelFClr; 2]; Clr[SelFClr; 3]; Clr[SelFClr; 4];
                        FillPercent;; 255; 255; 255; 100)
                Else
                If (InParagraph Or InTextBox)
                        FillColors (; Clr[SelFClr; 2]; Clr[SelFClr; 3]; Clr[SelFClr; 4]; FillPercent;;
                        255; 255; 255; 100)
                EndIf EndIf
        EndIf
EndIf
                //      NOW SET THE OTHER OPTIONS
        If (InTable)
                If (CntrTxt = 1)
                        TableCellJustification (Center!)
                EndIf
```

```
                If (LockCell = 1)
                        TableCellLock (Yes!)
                EndIf
                If (HeaderRow = 1)
                        TableHeader (Yes!)
                EndIf
        EndIf
                //      NOW DO WHATEVER CLEANUP IS NECESSARY
        If (InParagraph)
                TextBorderEnd (Save!)
                If (EndOfDoc)
                        PosDocBottom ()
                        PosLineUp ()
                        PosLineEnd ()
                        DeleteCharNext ()
                EndIf
                PosWordNext ()
        Else
        If (InTextBox)
                BoxUpdateDisplay ()
                BoxEnd (Save!)
                BoxEdit (TextBoxNumber)
                EditCut ()
                BookmarkFind ("Heading Macro Place Holder")
                BookmarkDelete ("Heading Macro Place Holder")
                EditPaste ()
                CloseGraphicsControlBar ()
        EndIf EndIf
        If (CodesOn)
                RevealCodes (On!)
        EndIf
        SelectMode (Off!)
EndIf

Label (End@)
StatusPrompt ("Off")
WaitMsgHide ()
WaitMsgDestroy ()
Quit

//************************************************************
//*      PROCEDURE: StatusPrompt
//*      INPUT: StatusPrompt
//*      OUTPUT: None
//*      DESCRIPTION: Displays a macro status prompt if not in a substructure.
//************************************************************
```

```
Procedure StatusPrompt (StatusPrompt)
If (Not ?Substructure)
        If (StatusPrompt = "Off")
                MacroStatusPrompt (Off!)
        Else
                MacroStatusPrompt (On!; StatusPrompt)
        EndIf
EndIf
Return
EndProc
//*****************************************************************

//*****************************************************************
//*      ROUTINE: Declarations@
//*      INPUT VARIABLES: None
//*      OUTPUT VARIABLES: Clr[]; Pat[]
//*      DESCRIPTION: Initializes arrays for color and pattern.
//*****************************************************************
Label (Declarations@)
Declare (Clr[DefinedColors; 4])
Clr[1; 1] := "White"
Clr[1; 2] := 255
Clr[1; 3] := 255
Clr[1; 4] := 255

Clr[2; 1] := "Black"
Clr[2; 2] := 0
Clr[2; 3] := 0
Clr[2; 4] := 0

Clr[3; 1] := "Dark Blue"
Clr[3; 2] := 0
Clr[3; 3] := 0
Clr[3; 4] := 127

Clr[4; 1] := "Dark Green"
Clr[4; 2] := 0
Clr[4; 3] := 127
Clr[4; 4] := 0

Clr[5; 1] := "Dark Cyan"
Clr[5; 2] := 0
Clr[5; 3] := 127
Clr[5; 4] := 127

Clr[6; 1] := "Dark Red"
```

```
Clr[6; 2] := 127
Clr[6; 3] := 0
Clr[6; 4] := 0

Clr[7; 1] := "Dark Magenta"
Clr[7; 2] := 127
Clr[7; 3] := 0
Clr[7; 4] := 127

Clr[8; 1] := "Dark Gray"
Clr[8; 2] := 127
Clr[8; 3] := 127
Clr[8; 4] := 127

Clr[9; 1] := "Brown"
Clr[9; 2] := 127
Clr[9; 3] := 127
Clr[9; 4] := 0

Clr[10; 1] := "Light Gray"
Clr[10; 2] := 192
Clr[10; 3] := 192
Clr[10; 4] := 192

Clr[11; 1] := "Light Blue"
Clr[11; 2] := 0
Clr[11; 3] := 0
Clr[11; 4] := 255

Clr[12; 1] := "Light Green"
Clr[12; 2] := 0
Clr[12; 3] := 255
Clr[12; 4] := 0

Clr[13; 1] := "Light Cyan"
Clr[13; 2] := 0
Clr[13; 3] := 255
Clr[13; 4] := 255

Clr[14; 1] := "Light Red"
Clr[14; 2] := 255
Clr[14; 3] := 0
Clr[14; 4] := 0

Clr[15; 1] := "Light Magenta"
Clr[15; 2] := 255
```

```
Clr[15; 3] := 0
Clr[15; 4] := 255

Clr[16; 1] := "Yellow"
Clr[16; 2] := 255
Clr[16; 3] := 255
Clr[16; 4] := 0

Declare (Pat[DefinedPatterns; 2])

Pat[1; 1] := "< None>"
Pat[1; 2] := 127

Pat[2; 1] := "100% Fill"
Pat[2; 2] := 9

Pat[3; 1] := "90% Fill"
Pat[3; 2] := 8

Pat[4; 1] := "80% Fill"
Pat[4; 2] := 7

Pat[5; 1] := "70% Fill"
Pat[5; 2] := 6

Pat[6; 1] := "60% Fill"
Pat[6; 2] := 5

Pat[7; 1] := "50% Fill"
Pat[7; 2] := 4

Pat[8; 1] := "40% Fill"
Pat[8; 2] := 3

Pat[9; 1] := "30% Fill"
Pat[9; 2] := 2

Pat[10; 1] := "20% Fill"
Pat[10; 2] := 1

Pat[11; 1] := "10% Fill"
Pat[11; 2] := 0

Pat[12; 1] := "Button Fill"
Pat[12; 2] := 10
Return
```

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

//***************************************
//      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 ()
DialogUndisplay ("WaitMsg"; "WaitText")
ENDPROC

//***************************************
//      PROCEDURE: WaitMsgDestroy
//      PURPOSE: Destroys a previously defined wait message dialog.
//***************************************
PROCEDURE WaitMsgDestroy ()
DialogDestroy ("WaitMsg")
ENDPROC
//*************************************************************
```