

# Polisher



## *Contents*

- ☞ Program Overview
- ☞ Main Screen and Controls
- ☞ File Menu
  - ☞ Project Dialog
- ☞ View Menu
- ☞ Process Menu
- Options ...
  - ☞ General
  - ☞ Indentation
  - ☞ Files
  - ☞ Spell Checker
    - ☞ Spelling Utility
  - ☞ White Space
  - ☞ Comments
  - ☞ Comment Blocks
    - ☞ Comment Block Definitions
- ☞ Troubleshooting - Technical Support
- ☞ License Agreement and Limited Product Warranty
- ☞ Product Registration
- ☞ Help Menu

# Program Overview:



***Polisher* (pòl'îsh'er) as a verb; To arrange **your** code, textbook perfect, every time - automatically!**

AARDVARK Software thanks you for your interest in *Polisher*.

*Polisher* is 100% Visual Basic® code, and demonstrates the enormous power Visual Basic brings to the Windows® developer. *Polisher* adds to Visual Basic's strengths by automatically formatting code files. Now, the individual programmer or a project manager can professionalize the *look and feel* of their source code. Using *Polisher*, output files will have a code structure, and attain a level of standardization that would please the most meticulous programmer. Project managers may establish guidelines for formatting code and comments with the expectation of those standards being *easily* maintained.

*Polisher* gives you a wide choice of formatting options to professionalize and standardize your code output. You may set the indentation, and the number and placement of blank lines for any code structure or comment style. Multiple line declarations may be split - listing each declaration on its own line, while the As <type> statements are aligned in a column of your choosing. Single line If statements may be split into a If block structure. Code instructions such as Option Explicit may be inserted, and Next variables removed. Another big plus, *Polisher* will spell check the input and output files, based on your pre-defined preferences.

While *Polisher* was designed for use with Visual Basic 3.0, it will produce *polished* code for other versions of Visual Basic - and other Basic dialects. In that regard, the information in this help file presumes the reader is knowledgeable of the Windows and Visual Basic standard conventions. If you are new to either, we strongly recommend you become familiar with the appropriate user's guide(s).

*Polisher* formats ASCII text files. *The Visual Basic Programmer's Guide* outlines the very real advantages of saving your code files in text format, and is worth reading. To get your code into text format do the following:

- (1) For a new project, go to the Visual Basic Options Menu - Environment Dialog Box. Select the option to save all of your project's code in text format.
- (2) To convert existing code files saved in binary format, go to the Visual Basic File Menu, and select the Save File As entry. When the dialog box appears, check the Save as Text box, then save the file using standard Windows conventions.
- (3) If you do not want to convert all of your code files to text format, you may use the File Menu - Save Text menu item to save a copy of any individual file.

## **Getting Started:**

To format a chosen project or file, open *Polisher*. After the *About Notice* has cleared the screen, click on the File menu and select the 'Open' menu item - or click on the toolbar 'Open' button. An Open File Common Dialog Box will be displayed. Using standard Windows conventions, select the project or file you wish to load, then click the 'OK' button. If you have selected a individual input file, it will be displayed in the main display area. If you have selected a project, the Project Dialog will be displayed. From the 'Project Dialog', select the files to be reformatted, then click on the 'Process All Selected Files' button. The first input file will be displayed in the main display area.

To immediately reformat the displayed input file- using the program's default settings, click on the 'Process' menu and select 'Polish' - or click on the 'Polish' toolbar button. The reformatted output file will now occupy the main display area. Save the *polished* file, and that's all there is to it!

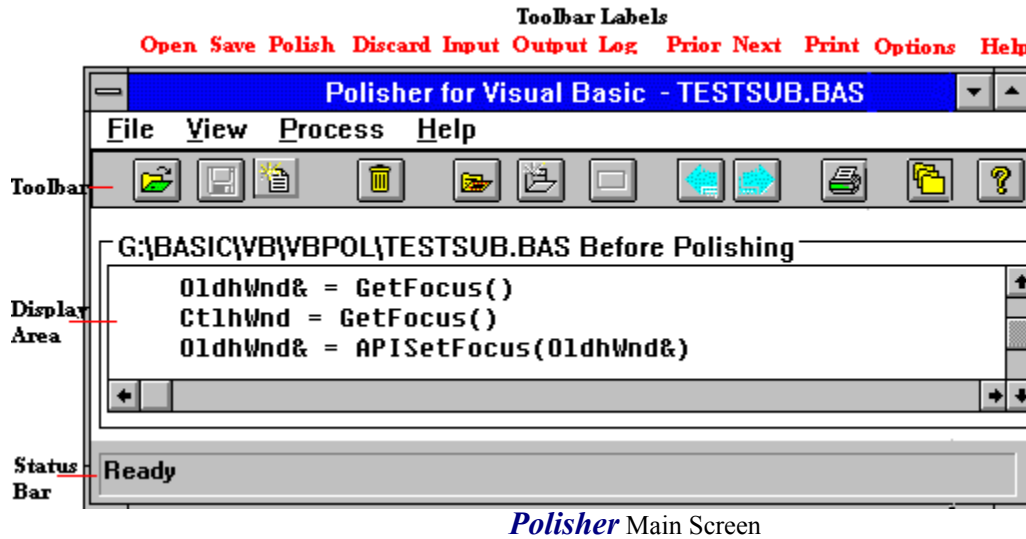
To refine the formatted output, click on the program's 'Option' menu; or *Polish* toolbar

button. The 'Option' tabs will be displayed. Each tab provides the user with the opportunity to define specific output styles. To get a better understanding of these, and other features supported by *Polisher*, use the Help Search Utility, the Contents screen for this file, or press F1 for context-sensitive help.



Visual Basic, Windows and Microsoft are registered trademarks of the Microsoft Corporation. Aardvark Software, Pretty Printer and Polisher are a trademarks of Aardvark Software, Inc.

# Main Screen and Controls:



*Polisher* Main Screen

**Title Bar:** Displays the application name at the top of a window. The Minimize and Maximize buttons are located to the right of the Title Bar. The Control Box (System Menu) is located to the left of the Title Bar.

**Menu Bar:** Displays the application's menus across the top of the window, below the title bar. (File, View, Options, Process, Help)

**Toolbar:** Located below the menu bar, the toolbar buttons give you quick mouse access to many of *Polisher's* commands and features as listed below:

**Labels (Toolbar Help):** Place the mouse cursor over a toolbar control. A label will be displayed informing you of the name of the control. To turn off the label display, click on the View menu, and uncheck the 'Toolbar Help' entry.

**Open:** Click on this toolbar button to display the Open Common Dialog Box. The default extensions are 'BAS', 'FRM' and 'MAK'. Using standard Windows conventions, select the project or file(s) you wish to load. If you have selected a project, the Project Dialog will be displayed. From the 'Project Dialog', select the files to be reformatted, then click on the 'Process All Selected Files' button. The first file will be displayed in the main display area. A single code files may be opened, or a multi-filed project may be loaded at any given time.

**Save:** Click on this toolbar button to save the reformatted output data to file. See Files tab to set backup options.

**Discard:** Click on this toolbar button to clear the display area, and close the input file. Any reformatted output will be discarded.

**Polish:** Click on this toolbar button to reformat the input file based on user selected and / or default options. During the process a gauge will be displayed on the screen. The gauge shows the time it takes to process the input file, while the Status Bar provides a running total of the number of lines reformatted. The gauge also displays the path of the input file.

**View Input File:** Click on this toolbar button to display the input file. This toolbar button is not enabled until a input file is loaded.

**View Output File:** Click on this toolbar button to display the output data. This toolbar button is not enabled until a input file is loaded and reformatted.

**View Log File:** Click on this toolbar button to display the log file for the current output file. Each output file has an associated log file. The log file contains the number of input lines, the number of output lines, the date, time, user, company, program header, and the output path. The log file may also contain a list of WARNINGS; which indicate areas requiring inspection or

manual adjustment. If you reformat an output file, the old log file is discarded, and a new file is written.

**Prior File:** If a project (\*.MAK) has been loaded, and multiple file have been selected for processing, click on this button (<) to view previously processed output files, if any. Otherwise this control is dimmed.

**Next File:** If a project (\*.MAK) has been loaded, and multiple file have been selected for processing, click on this button (>) to view the next input file to be processed. Otherwise this control is dimmed.

***Pretty Printer:*** Loads AARDVARK software's code printing utility with the current file or project. The first time you run ***Polisher***, a dialog box will be displayed requesting the path to VBPP.EXE. This information is stored in VBPOL.INI. If you acquire ***Pretty Printer*** at a later date, or change the print utility's disk location, clicking on this menu item will cause that same dialog box to be displayed again.

**Set Options:** Click on this toolbar button to display the Option Tabs. (General, Comment Blocks, Comments, Indentation, White Space, Files, Spell Check)

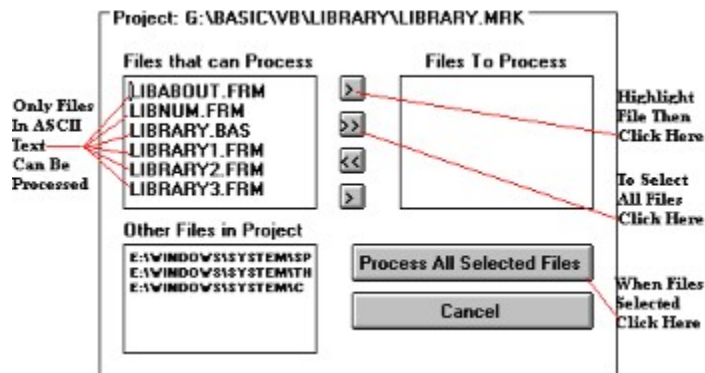
**Help:** Click on the Help button to display the 'Table Of Contents' for this file.

**Display Area:** The display area is the application screen space between the toolbar and the status bar where input and output code files are displayed.

**Status Bar:** Displays information and messages at the bottom of the application window that identify controls and their functions.

**Horizontal and Vertical Scroll Controls:** Horizontal and a vertical scroll bars are used for moving vertically and horizontally through the main display area with a mouse. Scroll bars are located at the right and bottom edges of the application window.

## Project Dialog:



*Polisher* Project Dialog

**Project:** The path and file name (\*.MAK) of the loaded project is displayed across the top of the 'Project Dialog'.

**Files In Project That Can Be Processed:** Project files listed in this box are ASCII text files. *Polisher* can only process files in ASCII text format.

**Files In Project To Be Processed:** Select individual files to be reformatted by highlighting the file name(s) in the 'Files That Can Be Processed' box, then click on the right (>) arrow button. Those file names will be copied to the 'Files In Project To Be Processed' box. Select all eligible files to be reformatted by clicking on the right double (>>) arrow button. To remove individual files from the 'Files In Project To Be Processed' box, highlight the file name(s) then click on the left (<) arrow button. To remove all files from the 'Files In Project To Be Processed' box, click on the left double (<<) arrow button.

**Other Files In Project:** The files listed in this box are support files (i.e., \*.VBX) or code files saved in binary format.

**Process All Selected Files:** After selecting the project file(s) to be processed, click on the this button to display the first input file in the main display area.

**Cancel:** Closes the 'Project Dialog,' and the loaded project without processing.

## File Menu:



**Open:** Select this menu item to display the Open Common Dialog. The default extensions are 'BAS', 'FRM' and 'MAK'. Using standard Windows conventions, select the project or file(s) you wish to load. If you have selected a project, the Project Dialog will be displayed. From the 'Project Dialog', select the files to be reformatted, then click on the 'Process All Selected Files' button. The first file will be displayed in the main display area. A single code file may be opened, or a multi-filed project may be loaded at any given time.

**Save:** Select this menu item to save the reformatted output code to file. If the output code has not previously been saved, then the Save Common Dialog will be displayed. Using standard Windows conventions, assign the path and name of the file to be saved.

**Save All:** Select this menu item to save all reformatted output code to file(s). If the output code has not previously been saved, then the Save Common Dialog will be displayed. Using standard Windows conventions, assign the path and name of the file(s) to be saved.

**Save As:** Select this menu item to display the Save Common Dialog. Using standard Windows conventions, assign the path and name of the file to be saved.

**Close:** Select this menu item to close the displayed file. If the output code has been changed since it was last saved, you will be asked if you want to discard the output file. Click 'Yes' to discard the file. Click 'NO' to continue, and save the file.

**Close All:** Select this menu item to close the open file and associated project. If the file has been changed since it was last saved, you will be asked if you want to discard the output file. Click 'Yes' to discard the file. Click 'NO' to continue, and save the file.

***Pretty Printer:*** Select this menu item to load AARDVARK software's code printing utility with the current file or project. The first time you run *Polisher*, a dialog box will be displayed requesting the path to VBPP.EXE. This information is stored in VBPOL.INI. If you acquire *Pretty Printer* at a later date, or change the print utility's disk location, clicking on this menu item will cause that same dialog box to be displayed again.

**Exit:** Select this menu item to end the *Polisher* session, and closes this help file if open. If the output code has been changed since it was last saved, you will be asked if you want to discard the output file. Click 'Yes' to discard the file. Click 'NO' to continue, and save the file.

## View Menu:



**Input Source:** This menu item is dimmed until a input file is loaded. After a file has been loaded, check this menu item to display the code file you wish to reformat.

**Output Source:** This menu item is dimmed until a input file is loaded and reformatted. After a file has been loaded and reformatted, check this menu item to display the output file.

**Log File:** Clicking on this menu item will cause the log entry for the current output file to be displayed. Each output file has an associated log. The log entry contains the number of input lines, the number of output lines, the date, time, user, company, program header, and the output path. The log entry may also contain a list of WARNINGS; which are those formatting tasks *Polisher* was unable to complete. If you reformat an output file, the old log entry is discarded, and a new entry is written.

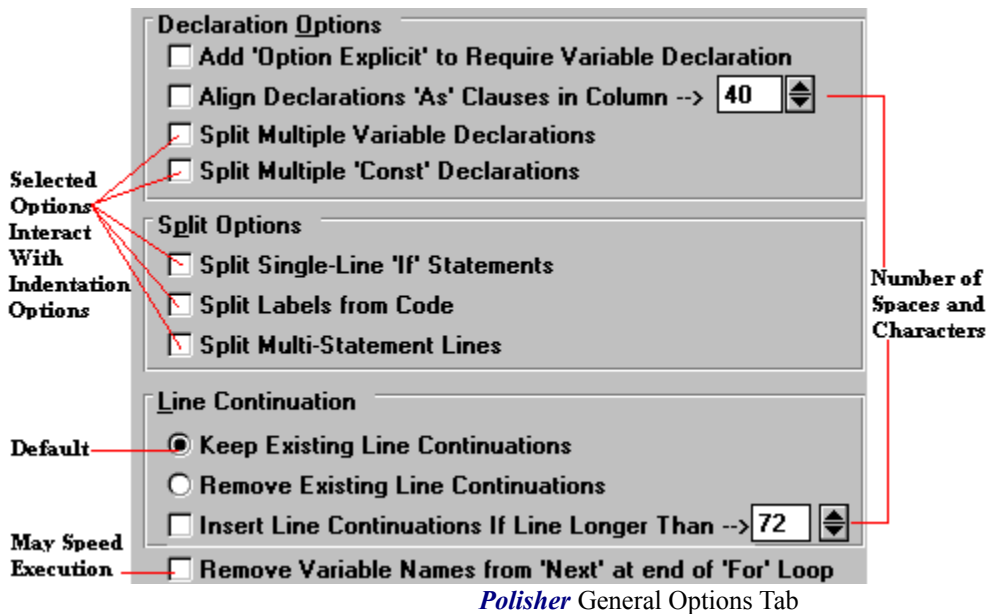
**Toolbar:** Check (default) - uncheck this menu item to display - hide the toolbar.

**Toolbar Help:** Place the mouse cursor over a toolbar control. A label will be displayed with the name of the control. To turn off the label display uncheck the 'Toolbar Help' entry.

**Status Bar:** Check (default) - uncheck this menu item to display - hide the status bar.



# Options - General:



**Add 'Option Explicit' To Require Variable Declarations:** *Polisher* places the **Option Explicit** statement in the General - Declarations section of a form or module. Use of the **Option Explicit** option is considered good coding technique. However, when this statement is used, all variables must be specifically declared. If you have not previously declared all variables within your project, and you select this option, the Visual Basic compiler will generate a message for each instance where an undeclared variable exist.

**Align Declarations 'As' Clauses In Column →:** Select this option to have each 'As <type>' statement be entered in a specified column. The first column location may be user defined by entering the desired start point in the box provided. Each additional 'As <type>' statement would be place to the right of the pervious entry. When this option is selected in conjunction with the 'Split Multiple Declarations' option, the result would place each declaration on its own line - with the 'As <type>' statements in a single column. (Example One)

**Remove Variable Names From 'Next' at end of 'For' Loop:** The insertion of variables associated with the **Next** (i.e., **Next I**) statement in Visual Basic is an optional procedure. Removing the variable(s) may make your application run slightly faster; however, code readability may suffer.

**Split Multiple Variable Declarations:** Select this option to have each declaration be entered on a line of its own. The placement of the declaration lines in the output file is dependent on the Indentation options selected. When this option is selected in conjunction with the 'Align Declarations As Clauses In Column' option, the result would place each declaration on its own line - with the 'As <type>' statements in a single column. (Example One)

**Split Multiple 'Const' Declarations:** Select this option to have each **Const** / **Global Const** declaration be entered on a line of its own. The placement of the declaration lines in the output file is dependent on the Indentation options selected.

**Split Single Line 'If' Statements:** Select this option to have the single line **If** statement be reformatted into a block **If** structure. The placement of individual lines of the **If** structure in the output file is dependent on the Indentation options selected. (Example Two)

**Split Labels from Code:** (numError: **Resume Next**) By selecting this option, and using the preceding illustration, 'numError:' would be entered on one line, and **Resume Next** would be entered on the following line. The placement of these lines in the output file is dependent on the

Indentation options selected.

**Split Multi-Statement Lines:** (x = y: a = b: z = a) By selecting this option, and using the preceding illustration, x = y, a = b, and z = a would each be entered on their own line, one following other. The placement of these lines in the output file is dependent on the Indentation options selected.

**Line Continuations:** Some dialects of Basic allow line continuations. The following options are provided in an attempt to meet the formatting needs of those Basic dialects.

**Keep Existing Line Continuations:** Select this option to make no line continuation changes to the current format.

**Remove Existing Line Continuations:** Select this option to create a single line of code by removing the line continuation(s).

**Insert Line Continuations If Line Longer Than →:** Select this option to insert a line continuation(s) by entering the maximum code line length (number of spaces and characters). The default is seventy-two (72) spaces. To define the maximum code line length, click on the ↑ to increase or click on the ↓ to decrease the number of spaces.

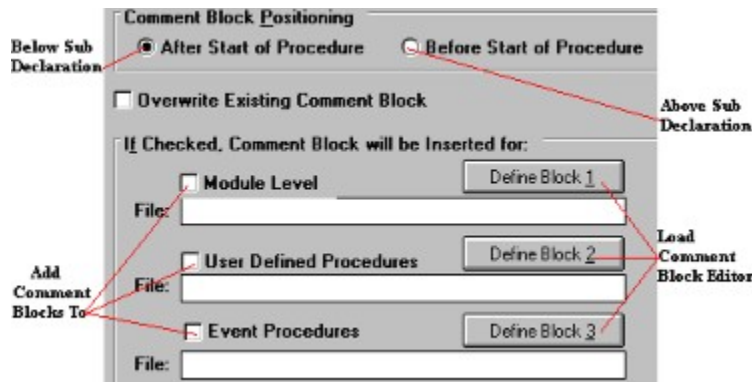
**OK Button:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.

VSVBX is copyrighted by Videosoft.

## Options - Comment Blocks:



*Polisher* Comment Block options Tab

### Comment Block Positioning:

**After Start of Procedure (default):** Select this option to insert Comment Blocks below the Sub / Function statement. To define the structure of the inserted Comment Block, click on the appropriate Define Block button.

**Before Start of Procedure:** Select this option to insert Comment Blocks above the Sub / Function statement. To define the structure of the inserted Comment Block, click on the appropriate Define Block button.

**Overwrite Existing Comment Blocks:** Select this option to replace existing Comment Blocks at their current position. To define the structure of the inserted Comment Block, click on the appropriate Define Block button.

### If Checked, The Comment Block Will Be Inserted For:

#### Module Level:

**Define Block:** To define the structure of the module Comment Block, click on the Define Block button.

**File:** If you have checked to insert a comment block, a file name must be specified. If a pre-defined module level Comment Block has been previously saved to file (\*.def), the path and file name has been saved in VBPOL.INI, and will be displayed in this block automatically. Otherwise click on the 'Define Block' button. *Polisher* will use the data from this file to insert module level Comment Blocks.

#### User Defined Level:

**Define Block:** To define the structure of the user defined level Comment Block, click on the Define Block button.

**File:** If you have checked to insert a comment block, a file name must be specified. If an existing user defined level Comment has been previously saved to file (\*.def), the path and file name has been saved in VBPOL.INI, and will be displayed in this block automatically. Otherwise click on the 'Define Block' button. *Polisher* will use the data from this file to insert module level Comment Blocks.

**Event Procedures:** *Polisher* considers routines to be event procedures if they are SUBs in a form module with a two part name divided by an underscore (xxxx\_yyyy).

**Define Block:** To define the structure of the event procedures Comment Block, click on the Define Block button.

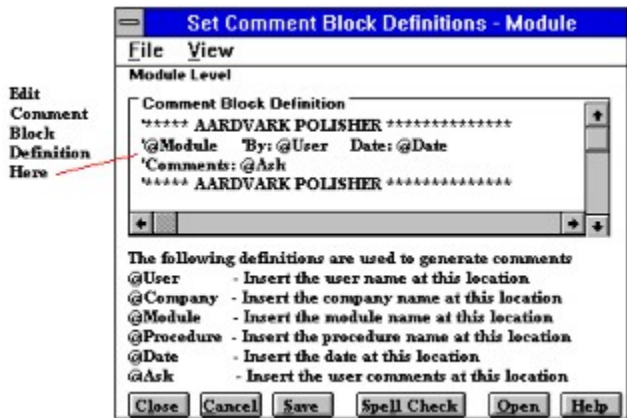
**File:** If you have checked to insert a comment block, a file name must be specified. If a pre-defined event procedure Comment Block has been previously saved to file (\*.def), the path and file name has been saved in VBPOL.INI, and will be displayed in this block automatically. Otherwise click on the 'Define Block' button. *Polisher* will use the data from this file to insert module level Comment Blocks.

**OK Button:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.

## Options - Comment Block - Definitions:



*Polisher* Comment Block Definition Editor

**General:** The default Comment Block Definitions for 'modules', 'user defined procedures' and 'event procedures' are contained in the file *cmtmod.def*. One of these default comment blocks will automatically be displayed in the Comment Block Definition window when the corresponding Define Block button is clicked. You may modify the displayed comment block by changing the text in the Comment Block Definition text window. To save your changes, open the 'File' menu, then click on 'Save' menu item.

### File Menu:

**Open:** If you have previously defined custom comment block, use this menu item to load the appropriate file. Selecting the 'Open' menu item causes the Open Common Dialog Box to be displayed. The default extension is 'DEF.' Using standard Windows conventions, select the file you wish to load.

**Save:** Selecting the 'Save' menu item will cause the redefined comment block to be saved to file. At the same time the path and file name of the saved file will be recorded in VBPOL.INI, and will be displayed in the 'File' entry of the Comment Block option tab automatically. If the data has not previously been saved, then the Save Common Dialog Box will be displayed. The default extension is 'DEF.' Using standard Windows conventions, assign the path and name of the file to be saved.

**Exit:** Closes the 'Comment Block Definition' text window. If the displayed comment block has been modified and not saved, you will be warned and asked: "Comment Block Definition Has Changed. Do You Want To Discard Changes?". Click 'No' to save modifications to file. If the data has not previously been saved, the Save Common Dialog Box will be displayed. Using standard Windows conventions, assign the path and name of the file to be saved. Click 'Yes' to close the 'Comment Block Definition' text window without saving modifications to file.

**View Menu:** Click on this menu to determine which 'Comment Block Definition' is being displayed.

### Comment Block Definitions:

**Module Comment Definition:** The default module level comment block is contained in the file *cmtmod.def*. The comment block may contain any desired text. The following are predefined elements, which may be edited to fit local requirements:

**User:** By default, *Polisher* inserts the registered Windows user's name in this space. In a custom comment block, the 'user' may be defined as the programmer, project manager, or any other individual so designated. You may also make this change in VBPOL.INI using any ASCII text editor. This file is located Windows\System sub-directory.

**Company:** By default, *Polisher* inserts the registered Windows company name in this

space. In a custom comment block, the 'company' may be defined as the client or the preparing organization. You may also make this change in VBPOL.INI.

**Module:** Insert the module name (i.e., EXAMPLE.BAS or FORM1.FRM).

**Procedure:** Module level comment blocks are placed at the beginning of the 'General Procedure' of the module or form.

**Date:** By default, *Polisher* inserts the date the program is run. In a custom comment block, the 'date' may be defined by client or company policy.

**Ask:** Insert user comments. 'Pre-defined User Comments' may include, or be made up entirely of statements based on company policy. During the reformatting process technical information relating to the procedure or module may be added.

**User Defined Procedures:** 'User Defined Procedures' are defined as any non-event procedure. The default user defined procedure comment block is contained in the file *cmtmod.def*. The comment block contains these elements:

**User:** By default, *Polisher* inserts the registered Windows user's name in this space. In a custom comment block, the 'user' may be defined as the programmer, project manager, or any other individual so designated. You may also make this change in VBPOL.INI using any ASCII text editor. This file is located Windows\System sub-directory.

**Company:** By default, *Polisher* inserts the registered Windows company name in this space. In a custom comment block, the 'company' may be defined as the client or the preparing organization. You may also make this change in VBPOL.INI.

**Module:** Insert the module name (i.e., EXAMPLE.BAS or FORM1.FRM).

**Procedure:** Insert the procedure name (i.e., SUB CENTERFORM).

**Date:** By default, *Polisher* inserts the date the program is run. In a custom comment block, the 'date' may be defined by client or company policy.

**Ask:** Insert user comments. 'Pre-defined User Comments' may include, or be made up entirely of statements based on company policy. During the reformatting process technical information relating to the procedure or module may be added.

**Event Procedures:** 'Event Procedures' are defined as any SUB in a form module with a two part name divided by an underscore (XXX\_YYY). The default event procedure comment block is contained in the file *cmtmod.def*. The comment block contains these elements:

**User:** By default, *Polisher* inserts the registered Windows user's name in this space. In a custom comment block, the 'user' may be defined as the programmer, project manager, or any other individual so designated. You may also make this change in VBPOL.INI using any ASCII text editor. This file is located Windows\System sub-directory.

**Company:** By default, *Polisher* inserts the registered Windows company name in this space. In a custom comment block, the 'company' may be defined as the client or the preparing organization. You may also make this change in VBPOL.INI.

**Module:** Insert the module name (i.e., EXAMPLE.BAS or FORM1.FRM).

**Procedure:** Insert the procedure name (i.e., FORM\_LOAD).

**Date:** By default, *Polisher* inserts the date the program is run. In a custom comment block, the 'date' may be defined by client or company policy.

**Ask:** Insert user comments. 'Pre-defined User Comments' may include, or be made up entirely of statements based on company policy. During the reformatting process technical information relating to the procedure or module may be added.

**Horizontal and Vertical Scroll Controls:** With a mouse, use the horizontal and vertical scroll bars to move vertically and horizontally through the Comment\_Block. The scroll bars are located at the right and bottom edges of the Comment Block Definitions text window.

#### **Form Controls:**

**Close:** Click on this button to close the Comment Block Definition text window. If the displayed comment block has been modified and not saved, you will be warned and asked: "Comment Block Definition Has Changed. Do You Want To Discard Changes?". Click 'No' to save modifications to file. If the data has not previously been saved, the Save Common Dialog Box will be displayed. Using standard Windows conventions, assign the path and name of the file to be saved. Click 'Yes' to close the 'Comment Block Definition' text window without saving modifications to file.

**Cancel:** Click on the 'Cancel' button to close the Comment Block Definition window without saving changes.

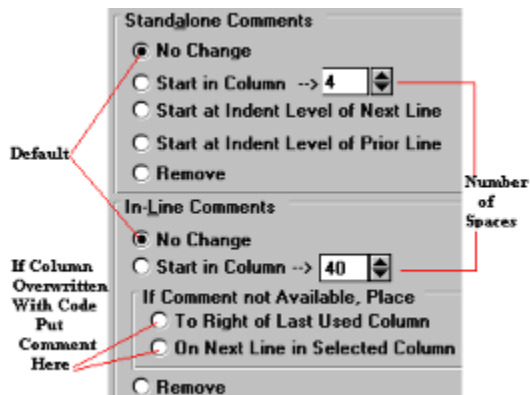
**Save:** Clicking on the 'Save' button will cause the redefined comment block to be saved to file. If the data has not previously been saved then the Save Common Dialog Box will be displayed. The default extension is 'DEF.' Using standard Windows conventions, assign the path and name of the file to be saved.

**Spell Check:** Click on this button to check the spelling of the displayed text.

**Find File:** If you have previously defined custom comment block, click on this button to find and load the appropriate file. An Open Common Dialog Box will be displayed. The default extension is 'DEF.' Using standard Windows conventions, find and select the file you wish to load.

**Help:** If you need help depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.

## Options - Comments:



*Polisher* Comments Option Tab

**Stand Alone Comments:** 'Stand Alone Comments' are comments that occupy their own line(s), separate from a code line.

**No Change (default):** Select this option if you want no change in the way the code file is currently commented.

**Start In Column (default) →:** Select this option to have each comment entry begin at a specified column. To set the initial column, click on the ↑ to increase or click on the ↓ to decrease the column number. The default is four. All 'Stand Alone Comments' will begin at this specified column.

**Start At Indent Level Of Next Line:** Select this option to have individual comment entries begin at the same indentation point as the following code line.

**Start At Level Of Prior Line:** Select this option to have individual comment entries begin at the same indentation point as the preceding code line.

**Remove Stand Alone Comments:** Select this option to delete 'Stand Alone Comments' from the output file.

**In Line Comments:** 'In Line Comments' are comments that follow the preceding code on the same line.

**No Change (default):** Select this option if you want no change in the way the code file is currently commented.

**Start in Column →:** Select this option to have each comment entry begin at a specified column. To set the initial column, click on the ↑ to increase or click on the ↓ to decrease the column number. The default is forty. 'In Line Comments' will begin at this specified column unless:

**If Column Not Available, Place:** If the column designated for entering 'In Line Comments' should be overwritten with code, select one of the following options:

**To Right of Last Used Column:** Select this option to insert 'In Line Comments' immediately after the end of the code line.

**Next Line in Selected Column (default):** Select this option to insert 'In Line Comments' on the first available line where the previously user defined column is available.

**Remove:** Select this option to omit 'In Line Comments' from the output file.

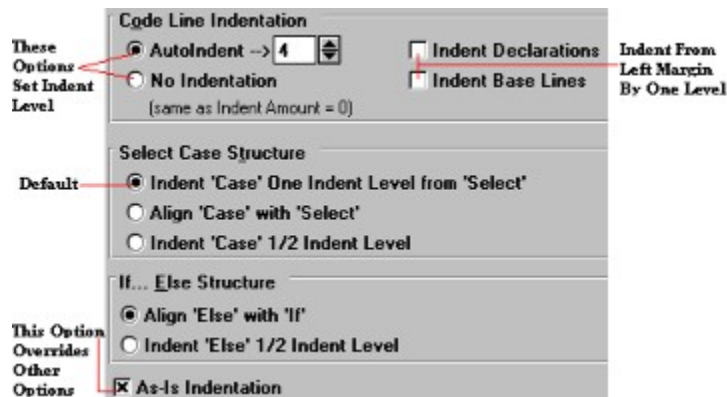
**OK Button:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.



# Options - Indentation:



*Polisher* Indentation Option Tab

## Code Line Indentation:

**Auto-Indent Amount (default) → :** Select this option if you want *Polisher* to automatically indent all code lines a specific amount. The default is four spaces. To set the 'Auto-Indent Amount', click on the ↑ to increase or click on the ↓ to decrease the number of spaces.

**No Indentation - Same As Indent Amount = 0:** Select this option if you want all code lines aligned to the left margin. This option supersedes the 'Auto-Indent' option.

**Indent Declarations:** (example: `Dim x`) Select this option to indent declarations one level (Auto-Indent Amount).

**Indent Base Lines:** Select this option to indent all code lines (as opposed to declaration lines) a minimum of one level. With this option selected, no executable code statements will be placed in column one unless Indent Amount = 0.

### Select Case' Structure:

**Indent 'Case' One Indent Level From 'Select' (default):** Select this option to indent **Case** statements one level (Auto-Indent Amount) to the right of the 'S' in **Select Case**.

**Align 'Case' with 'Select':** Select this option to align the 'C' in the **Case** statement with the 'S' in **Select Case**.

**Indent 'Case' ½ Indent Level:** Select this option to indent **Case** statements one half level (½ Auto-Indent Amount) to the right of the 'S' in **Select Case**.

### 'If...Else' Structure:

**Align 'Else' with 'If' (default):** Select this option to align the 'E' in the **Else** statement with the 'I' in the **If** statement.

**Align 'Else' ½ Indent Level:** Select this option to indent **Else** statements one half level (½ Auto-Indent Amount) to the right of the 'I' in the **If** statement.

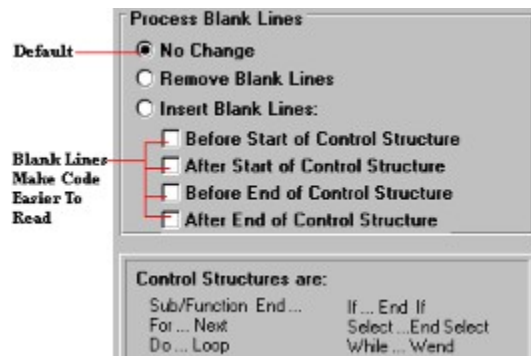
**As Is Indentation (default):** Select this option if you want no indentation changes made to the current format. This option supersedes all other indentation options.

**OK Button:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.

## Options - White Space:



*Polisher* White Space Option Tab

**No Change (default):** Select this option if you do not want to add or remove blank lines from the current format.

**Remove Blank Lines:** Select this option to remove all blank lines from the output file.

**Insert Blank Lines: Select this option to insert blank lines as indicated below:**

**Before Start Of Control Structure:** Select this option if you want to insert a blank line before each [Sub](#), [Function](#), [Do](#), [While](#), [For](#), [Select Case](#), and [If](#) statement.

**After Start Of Control Structure:** Select this option if you want to insert a blank line immediately after each [SUB](#), [Function](#), [Do](#), [While](#), [For](#), [Select Case](#), and [If](#) statement.

**Before End Of Control Structure:** Select this option if you want to insert a blank line before each [End Sub](#), [End Function](#), [Loop](#), [Wend](#), [Next](#), [End Select](#) and [End If](#) statement.

**After End Of Control Structure:** Select this option if you want to insert a blank line immediately after each [End Sub](#), [End Function](#), [Loop](#), [Wend](#), [Next](#), [End Select](#) and [End If](#) statement.

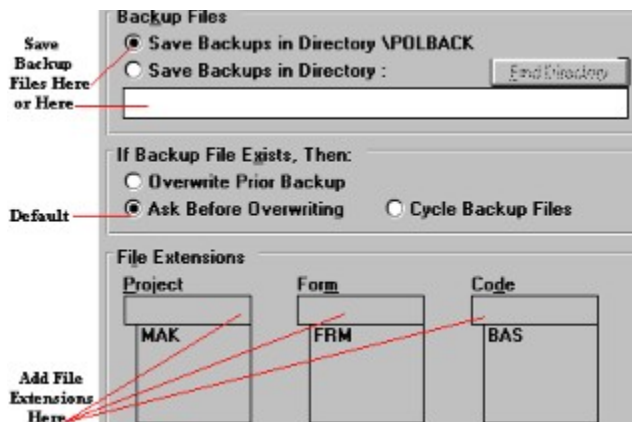
**OK Button:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.



## Options - Files:



*Polisher* Files Option Tab

**Backup Files:** By default, output files will be saved using the input file's name, extension and path. The following options allow the input files to be saved in their original format to designated sub-directory.

**Save Backup Files In Sub-Directory \polback (default):** Select this option to save input files in a sub-directory named *polback*. The *polback* sub-directory will be created as an extension of the input file's path.

**Save Backup Files In This Sub-Directory:** Select this option to save input files in a sub-directory of your choosing. You may enter the sub-directory's path directly in the space provided, or you may click the 'Find Directory' button. A Common Dialog Box will be displayed. Using standard Windows conventions, you may identify the sub-directory where the backup files are to reside.

**If Backup Files Exist, Then:** Each time a file is reformatted a backup file is generated. The following options determine whether backup files are maintained or overwritten.

**Overwrite Prior Backup:** Select this option, and each time a input file is reformatted, the previous saved backup file will be overwritten.

**Ask Before Overwrite (default):** Select this option, and each time a input file is reformatted, you will be asked if the previous saved backup file should be overwritten.

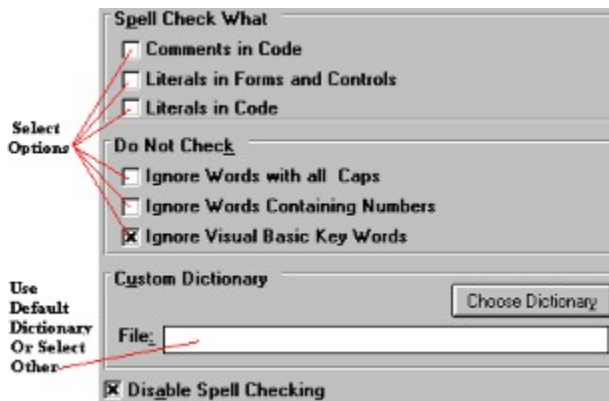
**File Extensions:** At the bottom half of the Files Tab three extension combo boxes are displayed. The listed extensions appear as defaults in the File Open Common Dialog Box. The default entries are 'BAS', 'FRM' and 'MAK'. You may assign new extensions. Example: Add the 'TXT' extension when Constant.txt is used in your project. You may add an extension by typing it in the combo box(es) provided, then depressing the Enter key. You may remove an extension by highlighting it with the mouse, then depressing the Delete Key.

**OK:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help' button. This file, open to the appropriate topic, will be displayed.

## Options - Spell Checker:



*Polisher* Spell Checker Option Tab

**Note:** Based on the options selected below - and the options selected in the spelling utility option dialog, code and comments are checked for correct spelling during the Polish (reformatting) process. Click on the "ABC" icon for instructions on the use of the spelling utility dialog



### Spell Check What:

**Comments In Code:** Select this option to check the spelling of the input / output file(s) comment entries.

**Literals In Forms and Controls:** Select this option to check the spelling of the input / output file(s) form and control level literal entries. Typical 'Form and Control Literals' are the 'Caption' and 'Text' properties. *Polisher* does not spell check words in the in the 'FontName' or 'LinkTopic' properties.

**Literals in Code:** Select this option to check the spelling of the input / output file(s) literal entries. *Polisher* processes alphanumeric and text literals in this category.

### Do Not Check:

**Ignore Words With All Caps:** Select this option if you want to disregard input / output file(s) capitalized words (i.e., EXAMPLE).

**Ignore Words Containing Numbers:** Select this option if you want to disregard input / output file(s) alphanumeric expressions (i.e., e9B7G).

**Ignore Visual Basic Key Words (default):** Click on this option if you want to disregard the spelling of the input / output file(s) key words.

**Custom Dictionary:** A custom dictionary may contain specialized words not found in the main spelling utility.

You can add words to the default custom dictionary (VBPOL.VTD) during the spell checking process; or if can identify another custom dictionary. Click on the 'Choose Dictionary' button to load the desired custom dictionary. A Common Open Dialog will be displayed. Using Windows standard convention, select the custom dictionary to be used (\*.VTD). The chosen dictionary will, until changed, be displayed in the space provide, and its path and file name will be recorded in VBPOL.INI.

**Disable Spell Checking (default):** Clear this check box to have your code and comments checked for correct spelling.

**OK:** Click on the 'OK' button to save the user defined options, and close the Tab Dialog.

**Cancel:** Click on the 'Cancel' button to close the Tab Dialog without saving changes.

**Help:** If you need help with a specific Tab depress the F1 function key, or click on the 'Help'

button. This file, open to the appropriate Tab, will be displayed.  
Visual Speller is copyrighted by Visual Tools, Inc.

## Process Menu:



**Polish File:** Click on this menu item to reformat the currently displayed input file. Formatting will be based on previously defined preferences recorded at the 'Option' menu tabs.

**Next File:** After processing the current input file, click on this menu item to display the next input file to be reformatted. This menu item is dimmed if there is no additional input file to be processed.

**Spell Check:** Click on this menu item to check the spelling of the displayed text. What text is checked, and which spell check utility is used, is based on user defined and / or default options selected at the 'Options' menu, Spell Checker tab.

**Skip This File:** Click on this menu item to close the currently displayed input file, without producing a reformatted output file.

**Discard Output File:** Click on this menu item to clear the display area, and close the current input file. Any output file data will be lost unless previously saved.

**Options ...:** Click on this menu item to display the Option Tabs. (General, Comment Blocks, Comments, Indentation, White Space, Files, Spell Check)

# Troubleshooting - Technical Support:



## General Information:

AARDVARK Software offers a variety of support options to help you get the most from *Polisher* for Windows. This section summarizes these options. If you have a procedural question about *Polisher*, use this file's search utility to look for possible solutions. Also, using any ASCII text editor, open the README.TXT in the directory where VBPOL.EXE was installed to check out our New Features, System Requirements, Upgrading From Previous Versions, Compatibility Issues, and the Most Frequently Asked Questions about *Polisher*.

If you need technical assistance beyond what this help file, or what the README.TXT file can provide, contact us by:

CompuServe: GO AARDVARK, 24 Hours-A-Day.

Fax: 1-201-833-1216, 24 Hours-A-Day.

Voice: 1-610-524-2589, Monday to Friday, 9:00 AM to 6:00 PM Eastern Time Zone.

E-mail: 73173.1137@compuserve.com

When requesting assistance by e-mail or fax please let us know how to reach you with our reply. Also, please send the following information that we may serve you better. DOS manufacturer and version, Windows version, printer make and model, print driver name and version, screen mode (VGA, SVGA, resolution, colors, driver). If you think *Polisher* is causing GPFs, sending the Dr. Watson log record would be helpful.

## Overview of Program Logic:

Program flow is as follows:

Program Initialization. Start-up parameters are read from the VBPOL.INI file found in the Windows directory. If the file does not exist, default values are used, and a VBPOL.INI file will be created. When a file is 'opened', it is partially read to fill the display. As the display is scrolled, additional parts of the file are read or reread. When a file is selected for polishing, it is reread and parsed. Then a new file is created in the TEMP directory (not necessarily that name, but pointed to by TEMP environmental variable), the contents are generated by applying the formatting options to the parsed source. After then, viewing of the input, output or log is done same as looking at input. The log file is created as part of the formatting process. Good place to look if problems.

If unable to initialize the program, try renaming VBPOL.INI to VBPOL.INX. Then restart the program. If that solves the problem, we want the VBPOL.INX file, along with an explanation of the problem.

BASICLAN.KEY. This file consists of over six hundred words that must remain in upper case ASCII sequence. If the file is corrupted (out of sequence), processing is terminated. User modification of this file can cause unpredictable results.

When the user selects one or more files to process, the selected file(s) are opened sequentially. The file is validated as a Visual Basic text-mode file. Even if it fails validation, the user may elect to process it. (We want to know about any 'good' file that fails validation.) Records are read from the file until the display is filled (about 20 lines). Additional lines are read as the display is scrolled.

<p><b>Note:</b> Polisher rearranges your code, and there is always a small chance the reformatted program will not compile exactly the same as the original version. Sometimes adding or removing even a single space can affect the results. It is always a good idea to revalidate your program after processing. If you find a problem, please let us know.</p>
--



## **AARDVARK SOFTWARE INC. LICENSE AGREEMENT AND LIMITED PRODUCT WARRANTY**

Please read this Agreement. Your opening of the software packet (the "Software") or use of the Software will indicate your acceptance. If you do not agree with these terms and conditions, you should promptly return the complete package for a full refund.

AARDVARK provides this Software and licenses its use to you. You are responsible for selecting the Software to achieve your intended results and for the installation, use and results obtained from the Software. The Software, including its code, documentation, appearance, structure and organization, is a proprietary product of AARDVARK and is protected by copyright and other laws. Title to the Software, or any copy, modification or merged portion of the Software, shall at all times remain with AARDVARK.

This License entitles you to:

1. Use the Software only on a single computer for your own individual use. If you alternately use several computers (for example, a laptop and a desktop), you may install the Software on each of them as long as it is never used on more than one simultaneously.
2. Make a reasonable number of copies of the Software for backup or archival purposes.
3. Transfer the Software, together with this License, to another person, but only if the other person agrees to accept the terms and conditions of this Agreement. If you transfer the Software and the License, you must, at the same time, either transfer all copies of the Software and its documentation to the same person or destroy those not transferred.

### **Limited Warranty**

EXCEPT AS SPECIFICALLY SET FORTH IN THIS AGREEMENT, THE SOFTWARE IS LICENSED AND PROVIDED ON AN "AS IS" BASIS, WITHOUT ANY KIND OF WARRANTY, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

AARDVARK warrants that the Software will substantially perform the functions or generally conform to the Software's specifications published by AARDVARK. AARDVARK does not warrant that the functions contained in the Software will meet your requirements or that the operation of the Software will be entirely error free or appear precisely as described in the Software documentation.

### **Remedies and Liabilities**

If you are not completely satisfied with the Software, AARDVARK's entire liability, and your exclusive remedy, shall be that you may return the Software (including all copies and documentation you may have relating thereto) within 90 days of your purchase, and obtain a complete refund.

In no event will AARDVARK be liable for any damages, including lost profits, lost savings, or other direct, indirect, incidental or consequential damages, arising out of the use or inability to use the Software, even if AARDVARK or a dealer authorized by AARDVARK had been advised of the possibility of such damages.

### **General**

If any provision of this Agreement is held to be unenforceable, the enforceability of the remaining provisions shall in no way be affected or impaired thereby. This Agreement shall be governed by the laws of the State of New Jersey.

## Registering Your Software



AARDVARK Software thanks *you* for your interest in *Polisher*.



AARDVARK Software encourages you to register *Polisher*. As a registered user, you are ensured of:

- Product support
- Advance notification of new products
- Advance notification of product upgrades
- Price discounts for product upgrades

To register your software, fill in and mail the registration card in your *Polisher* package.

## Help Menu:



**Context-Sensitive Help:** If you need help with a specific form or tab while running *Polisher*, depress the F1 function key. This file, open to the appropriate topic, will be displayed.

**Contents:** Click on the Help, Contents (Alt+H, C) menu item to view the Table Of Contents for this file.

**Search for Help On...:** Click on the Help, Search For Help On... (Alt+S) menu item to call the Help Search Utility. For instructions on the use of the Search Utility see Help On Help below.

**Help On Help:** Click on Help, Help On Help (Alt+H, H) to call the Windows tutorial Help On Help file.

**About Menu:** Click on the Help, About (Alt+H, A) menu item to see *Polisher* program and copyright information.

# Polisher



Copyright 1995 by:

**A** **ARDVARK**  
**SOFTWARE, INC.**  
972 Sheffield Road  
Teaneck, NJ 07666

**File Format:**

*Polisher* formats ASCII text files. *The Visual Basic Programmer's Guide* outlines the very real advantages of saving your code files in text format, and is worth reading. To get your code into text format do one of the following:

For a new project, go to the Visual Basic Options Menu - Environment Dialog Box. Select the option to save all of your project's code in text format. To convert existing code files saved in binary format, go to the Visual Basic File Menu - Save File As dialog box, and save the file using standard Windows conventions. If you do not want to convert all of your code files to text format, you may use the File Menu - Save Text menu item to save a copy of any individual file for use with *Polisher*.

**Default Extensions:**

You may add to, or change the default file extensions by clicking on the Set Options toolbar button, or by selecting the Process Menu, Options ... menu item, then select the Files tab. To add an extension type it in the appropriate combo box, then depress the Enter key. You may remove an extension by highlighting it with the mouse, then depressing the Delete key.

**ASCII** (American Standard Code for Information Interchange):

An ASCII file's contents are (mostly) printable and viewable text, as opposed to a binary file which generally contains coded characters. As applied to Visual Basic, there are two methods of storing Visual Basic programs on disk. The binary format offers faster loading time (in the integrated development environment). However, if corrupted, there is no easy way to recover the file's data, and GPFs may result. The ASCII format allows accessibility by most text editors and word processors, as well accessibility by various tools that Microsoft and third party vendors produce, such as the Setup Wizard utility.

**Continuations:**

Continuations are used to break a line of code into smaller elements, and still have the code execute as intended. Microsoft Visual Basic for Windows versions 3.0 and below do not support code line continuations. However, some other Basic dialects use line continuations, and *Polisher* attempts to process these.

A code line continuation is marked by a space followed by the underscore character at the end of the broken code line.

**Example:****Without:**

```
mnuBlank.Caption = Str$(Date) + Space$(3) + Format$(Now, "hh:mm")
```


**With:**

```
mnuBlank.Caption = Str$(Date) + Space$(3) _  
+ Format$(Now, "hh:mm")
```





***Pretty Printer*** is AARDVARK Software's renowned print utility. Black and white output is crisp, neat, and very readable. Using a color capable printer, ***Pretty Printer*** will produce a code listing that highlights Visual Basic's keywords, user variables, comments, literals, and page headings - each in a different color. Whether you use a laser, ink jet or dot matrix, ***Pretty Printer*** will produce a listing you will be proud to display. For ordering information contact AARDVARK Software at 1-800-482-2742.

**Portrait Orientation:**  There are eighty columns on a portrait oriented letter size (8½ x 11 inches, 210 x 297 mm) sheet of paper

### Option, General - Example One:

#### Align Declarations as Clauses in Column →: 5

Original Text: `Dim numA As Integer, numLongB As Long`  
Reformatted Text: `Dim numA As Integer, numLongB As Long`

#### Split Multiple Variable Declarations

Original Text: `Dim numA As Integer, numLongB As Long`  
Reformatted Text:  
`Dim numA As Integer`  
`Dim numLongB As Long`

#### Align Declarations as Clauses in Column →: 10 and Split Multiple Variable Declarations

Original Text: `Dim numA As Integer, numLongB As Long`  
Reformatted Text:  
`Dim numA As Integer`  
`Dim numLongB As Long`

### Option, General - Example Two:

Original Text: `If aDay = True Then Day = "Monday"`

Reformatted Text:

```
If aDay = True Then  
    Day = "Monday"  
End If
```

Original Text: `If num = 4 Then y = 4 Else y = 6`

Reformatted Text:

```
If y = 4 Then  
    Day = 4  
Else  
    y = 6  
End If
```

Original Text: `If aDay = True Then A = 2: B = 3: C = 4`

Reformatted Text:

```
If aDay = True Then  
    A = 2: B = 3: C = 4  
End If
```

**Comment(s):** Comments are comprised of text entries, and you may use them to document your code. Comments have no affect on code execution. The **Rem** statement, or an single quotation mark ( ' - apostrophe) must proceed all comment entries. Comment(s) are commonly used for:

- To explain individual lines of code.
- To explain the working and expected results of **Loop**, **For**, **Select Case**, and **If** structures.
- To explain the working and expected results of individual routines (**Sub** or **Function** level).
- To explain basic concepts and program function (module or form level).

**Comment Blocks:** Comment Blocks provide header information that may be helpful in understanding the code process. They are often required by clients or company policies. The information contained in Comment Blocks might consist of the date, name of company, name of module, module function, last update, general comments, etc. Comment blocks have a structured format, which may be detailed or general in nature, and ornate or plain in design.

**Literals:** These are control coordinates, MsgBox\$ text, variable equivalents, etc.

**Key Words:** Words or phrases that define Visual Basic objects, functions, statements, methods, properties, and events. Examples: [Dim](#), [For](#), [InStr\\$](#), [Val](#), [Do](#), etc.



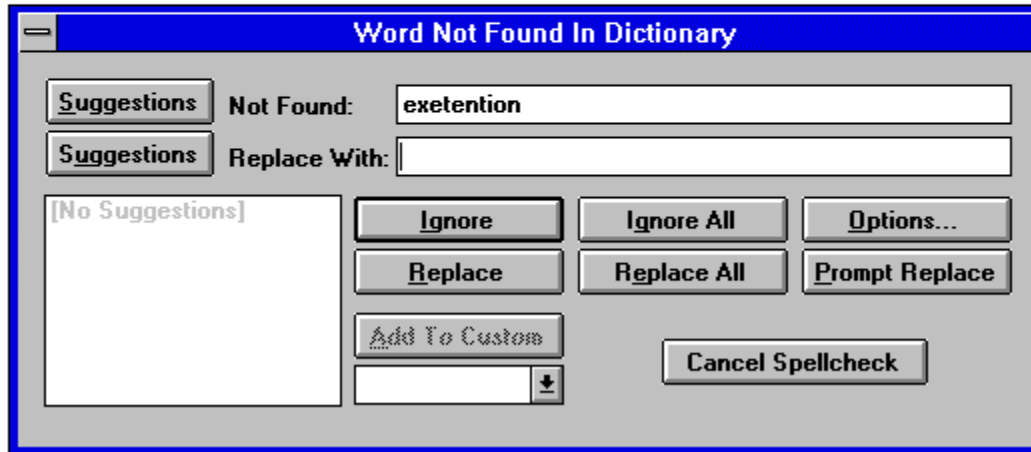
**Project File:** A project file (\*.MAK) is an ASCII text file that lists all the components of any Visual Basic Project, as well as the environmental options.

**Declarations:** Declarations include the following statements: [Dim](#), [ReDim](#), [Declare](#), [Option Base](#), [Set](#), [Type](#), and [Static](#).

**Parsing:** To break a code line down into its component parts, identifying declarations, keywords, literals, etc.

## Polisher's Spelling Utility - Dialog

```
Spell Checking the Comment>List program file exetention with reg.dat
'Call/close help file
Declare Function WinHelp Lib "User" ( ByVal hWwnd As Integer , ByVal lpHelpFile As String ,
>>List program file exetention with reg.dat <<
Declare Function RegSetValue Lib "SHELL.DLL" ( ByVal lpOpenKeyHand_ , ByVal SubKey
Declare Function RegCreateKey Lib "SHELL.DLL" ( ByVal lpOpenKeyHand_ , ByVal
```



**Spell Checking:** Spell checking is accomplished during the reformatting process. What type of terms are checked is based on the options selected at the Spell Check options tab, and the options selected in the spelling utility's option dialog. When the spelling utility finds a term it does not recognize, that term is displayed in the 'Not Found' area of the spelling utility dialog. The line of code where the term appears in the input file is displayed between the arrows (>> <<) in the dialog above the spelling utility display.

**Not Found:** When the spelling utility finds a term it does not recognize, that term is displayed in the 'Not Found' area of the spelling utility's dialog.

**Replace With:** The first suggestion generated by the dictionary is displayed here. If this word is the correct replacement, click on the 'Replace' button. If the first suggestion is not correct, double click on the correct suggestion. That term will then be displayed in the 'Replace With' area. If there are no suggestions generated, or the correct replacement word is not generated, type in the replacement in the space provided. When you are satisfied with the replacement term, click the 'Replace' button; or click the 'Replace All' button if the word to be replaced is repeated the code file.

**Suggestions:** When the spelling utility identifies a misspelled word, it also attempts to generate a list of suggested replacement terms. These suggestions are displayed in the box below the 'Suggestion' buttons. The first suggestion generated is also displayed in the 'Replace With' area of the spelling utility dialog.

**Ignore:** A word may be spelled correctly even though the spelling utility does not recognize the term. If this word is often used in your code files, you might consider adding the term to a custom dictionary. If the word is not likely to appear in other code files, click on the 'Ignore' button; or click on the 'Ignore All' button if the word is often repeated in this code file.

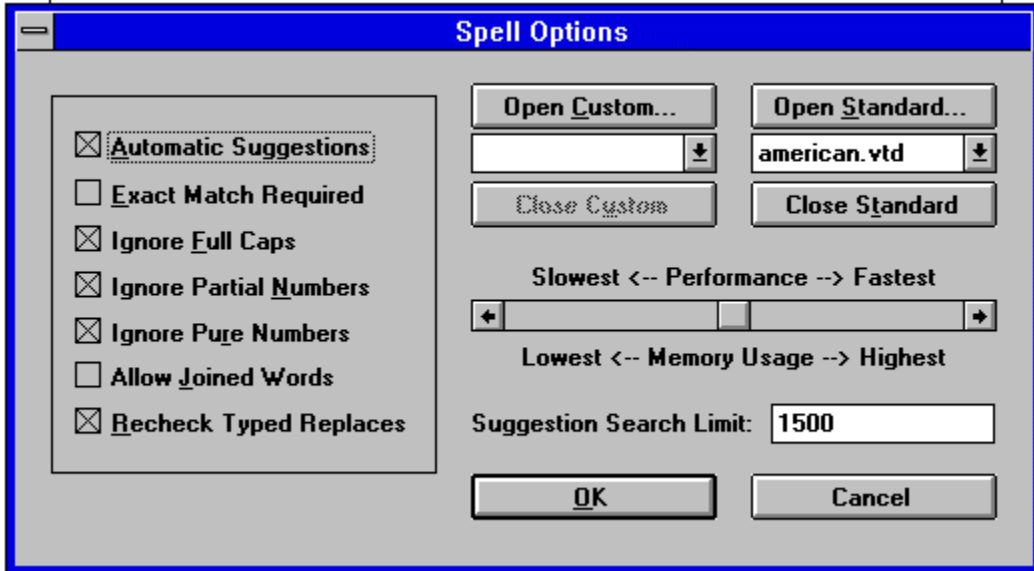
**Replace:** . When you are satisfied that the correct word in the 'Replace With' area, click the 'Replace' button; or click the 'Replace All' button if the word to be replaced is repeated the code file.

**Add To Custom Dictionary:** A custom dictionary may contain specialized words not found in the primary dictionary. If a custom dictionary is not listed, click on the options button to

install a custom dictionary.

**Options:** Click on this button to set the spelling utility's options.

**Cancel Spellcheck:** Click on this button if you wish to stop spell checking for the processing code file. A message will be displayed asking if you wish to suspend spell checking. Click 'No' if you wish only to skip this file, click 'Yes' if you want to disable spell checking for the duration of the session.



*Polisher* Spelling Utility Options Dialog

**Open Custom:** You can add words to the default custom dictionary (VBPOL.VTD) during the spell checking process; or if can identify another custom dictionary. Click on the 'Open Custom' button to load the desired custom dictionary. A Common Open Dialog will be displayed. Using Windows standard conventions, select the custom dictionary to be used (\*.VTD).

**Close Custom:** Click on this button to close the active custom dictionary.

**Open Standard...:** *Polisher* provides one standard dictionary, AMERICAN.VTD as its primary source for spell checking. Other dictionaries may be substituted. Click on this button to install a different primary dictionary.

**Close Standard:** Click on this button to close the primary dictionary. If no replacement is install, the spelling utility will use the custom dictionary to check spelling. Either a standard, custom or both dictionaries must be installed for the spelling utility to function.

**<--Performance -->:** Speed performance by clicking on the right (>) scroll bar arrow. Memory requirements increase as performance improves. Slow performance by clicking on the left (<) scroll bar arrow. This action might be required on systems with limited resources.

**Suggested Search Limit:** The number of words the spelling utility will compare in attempting to generate suggestions for an unrecognized term.

**Options:**

**Automatic Suggestions (default):** Select this option for the spelling utility to generate suggestions upon finding an unrecognized term.

**Exact Match Required:** Select this option for the spelling utility the compare terms in the processing file to its dictionary by case. (I.e., Example v. example: these two term would not be recognized as the same word.)

**Ignore Full Caps (default):** Select this option for the spell utility to disregard all capitalized terms (i.e., EXAMPLE).

**Ignore Partial Numbers (default):** Select this option for the spelling utility to disregard alphanumeric terms (i.e., 9Re8B).

**Ignore Pure Numbers (default):** Select this option for the spelling utility to disregard numeric expressions.

**Allow Joint Words:** Select this option for the spell utility to check hyphenated words (i.e., pre-basic).

**Recheck Typed Replaces (default):** Select this option for the spelling utility to check the spelling of words typed into the 'Replace With' area.

**OK:** : Click on this button to close the spelling utility options dialog and save changes.

**Cancel:** Click on this button to close the spelling utility options dialog without saving changes.

