PTVIEW Version 2.0

Contents of Program Text Viewer PTVIEW Help File

21 Oct., 1995

The Menu Items

File
Edit
Search
Processing
View
Window
Help

The **File** menu:

- **Open**: Open a file to the current view window. Use **Window | New Window** to open a new view window. Or you can put the line
  OpenWithNewWindow=T
  in the [GENERAL] section of your INI file to enable opening a new window for each file you open this way.
- **Reopen**: Re-open the current file on.
- **Save**: Save the current file.
- **Save As**: Save the current file with some other name.
- **Close**: Close the current file.
- **Close All**: Close all files of the focused view window.
- **Open INI File**: Open and use a user defined INI file. Usually, they are saved in the directory where PTVIEW is located, with the extension **ptv**.
- **Print**: Print the current file using the current view settings.
- **Print Setup**: Setup the printer.
- **Page Setup**: Setup print page layout.
- **Exit**: Exit PTVIEW.

The **Edit** menu:

- **Copy**: Copy the select text segment to the clipboard.
- **Select All**: Select the whole file.
- **Duplicate**: Duplicate the current file to other view window. Use **Window | New Window** to open a new view window.
- **Edit Line**: Edit the current line.
- **Insert Line**: Insert a line before the current line.
- **Add Line**: Add a line after the current line.
- **Delete Line**: Delete the current line.
- **Join Lines**: Join the the current and the next lines.

The **Search** menu:

- **Search**: Put the input focus to the search string edit box on the Search Bar. See Regular expression searching for more information on searching.
- **Search From Top**: Start the search from the top of current the file.
- **Search Forward**: Start the search from the next of the current line.
- **Search Backward**: Start the search from the previous of the current line, searching backward.

Go To: Put the input focus to the line number edit box on the Search Bar.
Go To Line 1: Go to the first line. That is to set the first line to be the current line.
Go To Current Line: Go to the current line, if it is scrolled off the screen.
Go To Line n: Go to the last line.
Go To Block Start: Go to the first line of the highlighted block nesting.
Go To Block End: Go to the last line of the highlighted block nesting.

The **Processing** menu:

Preprocess: Perform preprocessing. The most appealing features of PTVIEWrequires preprocessing. This function is good for top level program files; to preprocess any header files (included by some top level file), you must first preprocess the top level file, and then dive into it.

Bring Process to FG: Bring the preprocessing process to the foreground. If the process is running on the foreground, PTVIEW will not yield to Windows until the process is finished. Please note that you can press the *Esc* key to put a foreground process back to the background, to gain back control. If you prefer the process starts off foreground, add the line:
ProcessBackgroundInitially=F
in the [GENERAL] section of your INI file.

Cancel Process: Cancel the preprocessing process current running.

Browse Symbols: Popup the list of macro symbols, showing their definitions.

Free Symbol List: Free up the memory allocated for the symbol table after preprocessing. If you want PTVIEW to free the symbol table automatically after preprocessing, put the line
DeleteSymbolListAfterProcess=T
in the [GENERAL] section of your INI file.

Free Preprocess Info: Free the preprocess information retained. The preprocess information is necessary to highlight conditionally-compiled away code segments, and to show block nestings.

Set Directory: Set the working directory (the include directory). Also see preprocess settings dialog.

Preprocess Settings: Popup the preprocess settings dialog.

Convert: Convert tab characters using the new tab size specified, while keeping the look of the text file. Use the menu item **Processing | Tabs Settings** to set the tab sizes.

Tab Settings: Popup a dialog so that you can specify the current tab size, and the new tab size (for tab characters conversion).

The **View** menu:

Files: Popup the list of files opened for the focused view window. Choose the desired one to be the current file.

File List: Toggle the File List Bar.

Line Numberings: Toggle the line numbering bar, the vertical bar which shows the line numberings.

Block Nestings: Toggle the block nesting bar, the vertical bar which shows the C/C++ block nesting outlines. Note that preprocessing is necessary for PTVIEW to outline the

block nestings.

Highlight Comments: Toggle highlighting of C/C++ comments.

Show Tabs: Toggle showing the tab characters as rectangles.

Show Complete: Toggle showing conditionally-compiled awaytext lines. See preprocessing for more on conditional compilation.

Tool: Toggle the Tool Bar.

Search: Toggle the Search Bar.

Text Font: Popup a dialog for selecting font to use in all view windows (and printing).

Text Colors: Popup a dialog for selecting various text colors.

The **Window** menu:

New Window: Open a new view window. With an extra view window, you can duplicate an opened file from another view window. (Use the menu item **Edit | Duplicate**.)

Cascade: Cascade the view windows.

Tile Horizontal: Horizontally tile the view windows.

Tile Vertically: Vertically tile the view windows.

Arrange Icons: Arrange the view window icons.

Close All: Close all of the view windows.

The **Help** menu:

Contents: Bring up this help file, showing the contents of the help file.

Overview: Bring up this help file, showing the section which gives you an overview of PTVIEW.

Regular Expression: Bring up this help file, showing the section which talks about regular expression.

About: Popup the about dialog, telling you some information about the current state of PTVIEW.

To speedup selecting menu items, three speed bars are created:Tool Bar, Search Bar, and File List Bar.

The following is how the File List Bar looks like:

Click on this button to open new file for the view window.

This "on" button tells the name of the current file on the view window.
You may drag this button and drop it on any other view window (MDI child window) to duplicate the file on that other view window.

This "off" button tells the name of another opened, invisible file on the view window. Click on the button to select that file for viewing.

These empty "off" buttons are unused space for files.
Note that the number of opened files is not restricted to the number of file buttons on the File List Bar.

Use this scroll bar to shift the file list so as to see the hidden opened file names.

Click on this button to popup the list of opened file for selection.

Click on this button to cause the file-list bar to disappear.

The text file shown on the focused view window is the current file.

Running PTVIEW

Components:

PTVIEW.exe - The program executable itself.
PTVDEF.dll - A supporting DLL that deals with compiler system specific settings.
BWCC.dll - Another supporting DLL.
PTVIEW.hlp - This help file.

For simplicity, put all these files in the same directory; then, you should start PTVIEW from within that directory. If you are sure that there is already a copy of **BWCC.dll** in your Windows system directory, you can safely delete the copy that comes with the program.

Command Line:

PTVIEW [%<*file1*>] [<*file2*>]

- <*file1*> is an INI file. It also specifies the INI directory --- the directory of the program by default --- in which PTVIEW believes all INI files should be. If <*file1*> is a directory (ended with '\'), then it only specifies the INI directory.
- <*file2*> is a text file to view.

PTVIEW is developed with Windows 3.1 in mind. Therefore, it is a 16-bit Windows application. I have tried it on Windows 95 and Windows NT; it seems to work fine on both platforms.

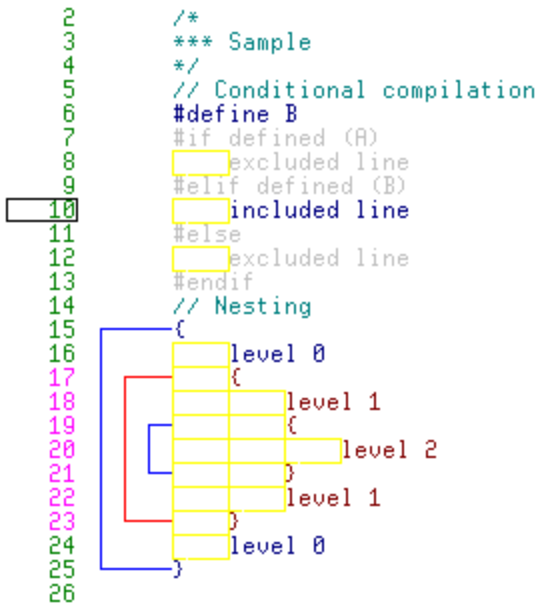
Overview of Program Text Viewer PTVIEW

PTVIEW version 2.0 is a small Windows 3.1 utility which aids viewing dazzling C/C++ program text files, which may possibly be clustered up by conditionally compiled code segments. It can hide/highlight the code segments which will not be compiled; it can outline statement blocks mangled among these segments; and it can even print them out to printer.

PTVIEW is designed with several pretty nice features:

- C/C++ preprocessing
- Printing
- Regular expression searching
- Tab size converting

The following is a snap-shot of the application's screen:



Note that the above coloring scheme can be customized.

The green numbers are normal line numbers.

The number with a rectangle drawn around it is the current line number. It tells you which line is the current line.
You can click on any other line numbers to set it to be the current one.

The red numbers are lines numbers of the highlighted (current) block text lines.

The blue lines outline the C/C++ block nestings.
Click on any one of such lines to highlight the outlined block.The highlighted (current) block nesting is outlined in red. To unhighlight, click while holding down the *Ctrl* or *Shift* key.
Note that preprocessing is necessary for PTVIEW to show the block nestings.

The dark blue text lines are normal --- not conditionally compiled away --- text line.
You can double click, while holding down the *Ctrl* key, on any macro symbol to find out the definition of that symbol at that line.
Note that preprocessing is necessary for PTVIEW to distinguish between different text lines, and to know the definitions of macros.

The gray text lines are conditionally compiled away text lines.
You can hide/show those lines with the menu item **View | Show Complete**, or checking/unchecking the **Show Complete** check box on the Tool Bar.

The dark green text lines/characters are C/C++ comments.
Note that PTVIEW highlights comments correctly only if they are not nested.

The dark red text lines are the lines of the highlighted (current) block.

There are three kinds of speed bars: Tool Bar, Search Bar, and File List Bar.

Searching with Regular Expression

One feature of PTVIEW is searching with regular expression. If you are already familiar with regular expression, you will have no problem performing search with PTVIEW. Otherwise, you either learn the syntax of regular expression and be happy with it, or you disable it altogether.

To perform searching, the Search Bar must be visible. If it is not currently visible, by selecting the menu item **Search | Search** the Search Bar is automatically made visible, and the cursor is put in the edit box where you enter search strings. Remember that a search string can be a regular expression if the **RE** check box on the Search Bar is checked. Once you get used to regular expressions, you will find them very helpful.

To start the search, you can just press *Enter* after entering the search string. This causes the search to start from the next of the current line. Alternatively, you can use the menu item **Search | Search Forward** to attain the same effect. If you use the menu item **Search | Search From Top**, you start the search from the top of the text file. If you use the menu item **Search | Search Backward**, you start the search from the previous of the current line, searching backward. The search pattern strings are memorized. You can walk through the list of memorized search strings using the up/down arrow keys. Please notice that the searching algorithm implemented only tries to spot whether a particular line contains the search pattern or not. The number of occurrences of the pattern within a line doesn't matter.

Another similar feature is going to a particular line. (Menu item **Search | Go To**). You enter the line number to go to in the leftmost edit box of the Search Bar. Then by pressing *Enter*, the current line will be set to the line you specified. (Actually, the current line jumps as you enter; however, PTVIEW memorizes the line number if you press *Enter*. You can use the up/down arrow keys to walk through the list of memorized line numbers.) You may also use the menu items **Search | Go To Line 1** and **Search | Go To Line n** to go to the top and bottom of the text file respectively.

The following is how the Search Bar looks like:

The regular expression syntax, which is very similar to that used by the UNIX VI editor, used by PTVIEW is described as follows:

A regular expression is composed of characters.Some are just regular characters with regular meanings; like 'a', 'A', 'b', 'B', '1', '2', ... While some have special meanings:

- **.** represents any character.
- **^** represents the beginning of a line.
- **$** represents the end of a line.
- **\<** represents the begin of a word.
- **\>** represents the end of a word.

[*char-list***]** represents any one of the characters from *char-list*; if *char-list* is prefixed by '^', the reverse is true.It is possible to represent a list of consecutive characters by using '-'.For example, **A-Z** represents the list of characters from 'A' to 'Z'inclusively.

\(and **\)** delimit a subexpression.

\t and **\b** are the familiar C/C++ control characters which Iadded to the syntax.

After a character (or one of the above special "characters"), a repeat count can be specified:

***** means 0 or more times.

\+ means 1 or more times.

\? means 0 or 1 times.

\{*n***\}** means exactly *n* times.

\{*n***,***m***\}** means *n* to *m* times.*m* can be omitted; in this case, *m* is assumed to be infinite.

The escape character is the usual '\'.Therefore, **\.** represents the character '.' (the dot), because the special meaning of '.' is escaped away.

Click on this button to cause the search bar to disappear.

Enter the line number you want to go to here. Pressing *Enter* will take you there.
Note that a few tens of the previous line numbers you entered are remembered. You can use the up/down arrow keys to walk through the list.
This is also where the current line number is displayed.

Check this box if you want your search be case-sensitive.

Check this box if you want the search string you entered be treated as a regular expression.

Enter the search string (a pattern) you want to search for in this box. Pressing *Enter* will start the search from the next of the current line.
Note that a few tens of the previously entered search string are remembered. You can use the up/down arrow keys to walk through the list.

Start searching for the pattern you specified from the top of the text file.

Start searching for the pattern you specified from the next of the current line, searching forward.

Start searching for the pattern you specified from the previous of the current line, searching backward.

A string of characters composed of digits and/or alphabets (including '_') only is considered to be a word.

Setting and Converting Tab Size

As a programmer, I believe that there were times you felt almost mad because the code you were reading was not properly aligned. The problem is usually that the original author of the file used some tab size, and you view it using another. In this situation, you may wish that you could quickly test out the different tab sizes to determine which one was used. Furthermore, you may even want to get rid of those confusing tab characters once and for all. (Or convert the tab size using your favorite one, if you yourself are used to using tab characters.) PTVIEW allows you to do exactly this. All you want to do can be controlled with the Tool Bar. (Actually, many functionalities of the Tool Bar are releated to preprocessing.) Or if you are used to using the menu, check out the Processing menu.
Please note that all trailing blanks/tab characters will be removed.

The following is how the Tool Bar looks like:

Check this box to make the line numberings visible.

Check this box to make the block nesting outlines visible.

Check this box to make tab characters represented by yellow rectangles.

Check this box to make conditionally compiled awaytext lines visible. See preprocessing for more on this.

Click on this button to perform preprocessing.
Preprocessing will enable you to visually distinguish conditionally compiled away text lines from the others.

Click on this button to popup the preprocess settings dialog.

Click on this button to popup the list of macro symbols.

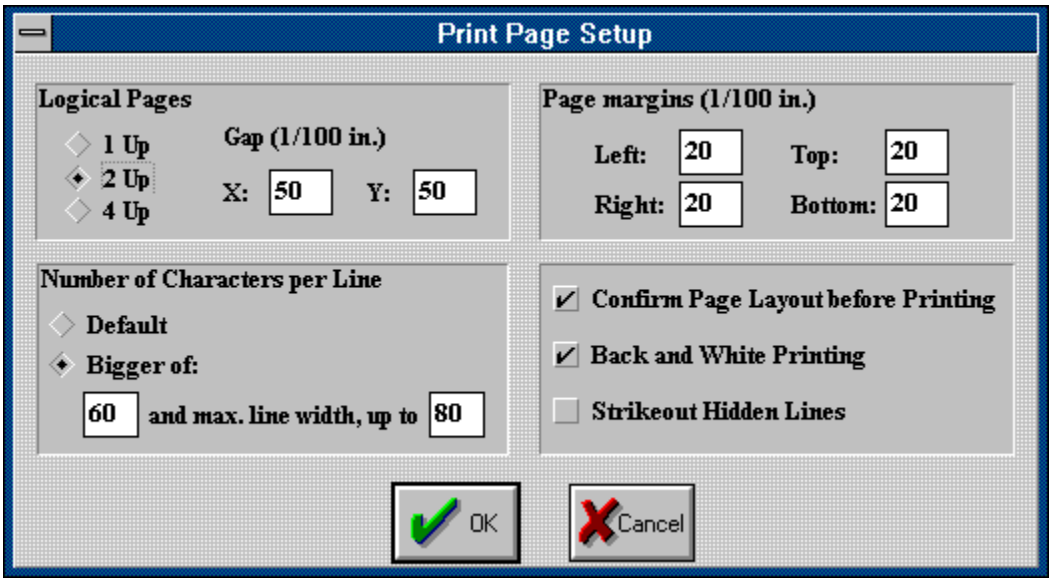
Use this scroll bar to adjust the desired tab size.

Click on this button to convert tab characters, while keeping the look of the file.

Click on this button to cause the tool bar to disappear.

Printing

The following dialog box, which sets the logical print page layout on a physical page, should show the printing features of PTVIEW:



Note that the above dialog box is activated with the menu item **File | Page Setup**.

Checking this radio button means that you want to print a single logical page per physical page.

Checking this radio button means that you want to print 2 logical pages per physical page.
If the physical page orientation is Portrait, i.e. page height is bigger than page width, one logical page will be on top of the other. Otherwise one logical page will be on the left side, and the other on the right.

Checking this radio button means that you want to print 4 logical pages per physical page.

Checking this radio button means that you want to print however many characters per line the select font allows. In other words, no scaling will be performed.

Checking this radio button means that you want to have some control over the number of characters printed per line. You specify the number of characters per line by filling the two edit boxes beneath; however, the accuracy depends on the resolution of your printer.

Enter the minimum number of characters per line in this edit box. Note that the value you entered here is significant only if the above radio button is checked.

Enter the maximum number of characters per line in this edit box. Note that the value you entered here is significant only if the above radio button is checked.

Checking this check box means that you want to be notified of the effective print layout before printing.

Checking this check box means that your printer is not capable of reasonably representing color, and you want the printout in pure black and white. Note that the background is never painted for black and white printing.

Checking this check box means that you want the hidden lines, if they are to be shown, be "crossed" out. Note that this options override any background color for hidden lines.

Enter the length (in hundredth of an inch) of the gap between two logical pages laid out horizontally here.

Enter the length (in hundredth of an inch) of the gap between two logical pages laid out vertically here.

Enter the physical page margin (in hundredth of an inch) here.

C/C++ Preprocessing

PTVIEW is capable of performing some simple C/C++ preprocessing:

1. It finds out and highlights conditionally compiled away code segments (those lines which will be ignored by the compiler). (See screen snap-shot.) After PTVIEW identified the conditionally compiled away code segments, you can choose to highlight them, or hide them altogether using the menu item **View | Show Complete**. Note that PTVIEW would have trouble evaluating macro with parameters --- **defined** works; **sizeof** doesn't --- or macro expressions which contains strings or floating point numbers. (By the way, all numbers are treated as long integers by PTVIEW; and a undefined symbol assumes a value of 0, and a "empty" defined symbol assumes a value of 1.)
As a side effect, you can find out the definition, and evaluated value if possible, of any macro symbol in your program text. You do this by double-clicking the symbol while holding the *Ctrl* key down. The definition, and evaluated value if possible, of the symbol at that particular line will be popped up.
Additionally, you will be presented with a list of file names, which indicates the path of file inclusions that leads to the definition of the symbol. Note that you can double click on any one of the file names to open (dive into) it, and be positioned at the line number involved.
Don't be alarmed if you see symbols like **ZZZZ>>>>**. They are defined by PTVIEW to mark the lines at which file is included.

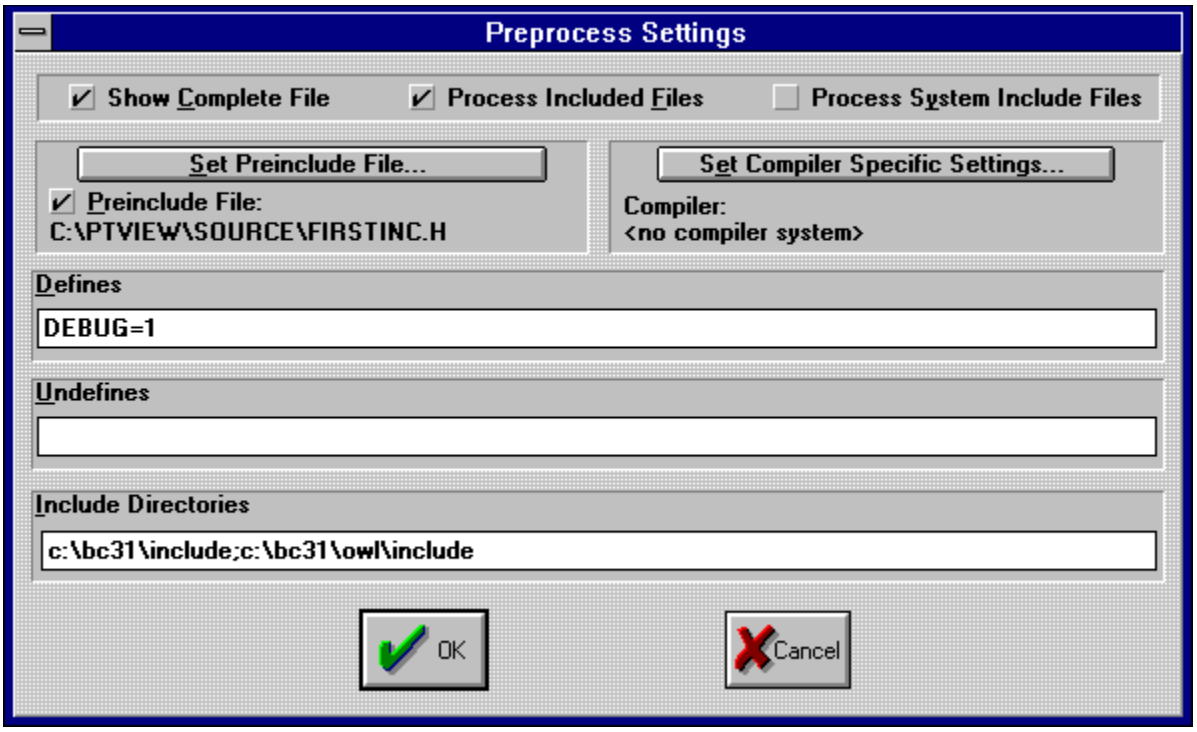
2. It finds out and outlines the C/C++ block nestings. (See screen snap-shot.) It gives you a visually easier way to identify code text lines that belongs to a particually block. Note that you can select a block by clicking on any block nesting outline in order to highlight the code text lines belonging to that block. You can deselect by clicking while holding down the *Ctrl* or *Shift* key.

Note:
1) PTVIEW cannot handle nested comments correctly.
2) PTVIEW cannot handle DBCS correctly.
3) PTVIEW uses a hash table to store macro symbol information. You can set the size of the hash table by adding the line

    SymbolHashTableSize=*n*

    in the [GENERAL] section of your INI file. The value *n* is bound by a minimum value of 2 and some maximum value; the default value is 107. Please be advised that *n* is NOT the number of symbols PTVIEW can store. Each entry of the hash table is used to store "similar" symbols, upto around a thousand symbols.

The following is the preprocess setting dialog:


Preprocess Settings
Show Complete File
Process Included Files
Process System Include Files
Set Preinclude File...
Preinclude File:
C:\PTVIEW\SOURCE\FIRSTINC.H
Set Compiler Specific Settings...
Compiler:
<no compiler system>
Defines
DEBUG=1
Undefines
Include Directories
c:\bc31\include;c:\bc31\owl\include
OK
Cancel

If the current file is preprocessed with symbol list kept around, you can dive into any include filed (open and preprocess it). To do this, one way is:

1. Set the current line to be the line which include the file you want to dive into.
2. Bring up the view window right mouse menu by right mouse clicking the view window.
3. Select the menu item **Dive into *file***.

Note that the preprocessing of include file is based on the information gathered from preprocessing the top level file.

Check this box if you want to display the conditionally compiled away text lines.

Check this box if you want PTVIEW to process also the user include files (e.g. #include "IncFile.h").
The include files will always be read from disk, even if it is opened. You set the path of the working directory with the menu item **Processing | Set Directory**. Please notice that you can specify additional include directories.

Check this box if you want PTVIEW to process also the system include files (e.g. #include <stdio.h>).
The include files will always be read from disk, even if it is opened. You set the path of the working directory with the menu item **Processing | Set Directory**. Please notice that you can specify additional include directories.

Preprocessing include files is significant only if the include files define macro symbols. Therefore, in some cases, it makes sense to just preprocess a single additional include file (instead of all the include files specified in the file). You set that single include file by clicking on the button.
Note that all include files specified in the pre-include file will always be preprocessed.

Check the box if you want to set a pre-include file.

The effective include file to pre-include is displayed here.

Enter the additional include directories, separated by semi-colors (;), here.
Note that the primary directory is set with the menu item **Processing | Set Directory**, which is the working/current directory. And the additional include directories will be relative to this working directory, if they are not absolute paths.

Click this button to set the compiler system you are working with. PTVIEW needs to know the compiler system you are working with so that it can pre-define the compiler specific pre-defined macro symbols for you automatically.
Currently, PTVIEW recognizes **Borland C++** version 3.x configuration files (you can convert a project file (**prj**) to a configuration file (**cfg**) using the utility **PRJCFG**), and **Microsoft Visual C++** version 1.5x make files only. For other compiler systems, you have to specify the pre-defined macro symbols manually.
Note that the settings are written to an INI file. The default INI file is "ptview.ini" in your Windows directory. You can specify a custom INI file with the menu item **File | Open Ini File**.

The effective compiler system's name is shown here.

Enter additional user pre-defined macro symbols here (e.g. DO_DEBUG=1;USE_LIB=0). This line is processed after processing the compiler specific defines.
Note that they should be separated with semi-colons (;).

Enter symbols to undefine here. This line is processed after processing the compiler specific defines and user defines; and is processed before any preprocessing of files.

Accept the changes made to the settings.

Cancel the changes made to the settings.

PTVIEW is designed and implemented by **Trevor W.S. Lee**.

Conditionally compiled away text lines refer to those lines that will be ignored by the compiler due to preprocessor conditions set. For example, if WIN32 is defined:

```
#if defined (WIN32)
// This line will be compiled.
#else
// This line will be ignored.
#endif
```

Text Editing

PTVIEW is equipped with a minimal set of editing capabilities. The idea is to allow you to edit a single line at a time, using a line edit control. You can start editing a line by pressing *Alt+E*. Once you are editing a line, you can expect some usual editor behaviours like moving the cursor in any direction by pressing array keys, splitting a line at the cursor position by pressing *Enter*. Pressing *ESC* will terminate the editing. Please refer to the Edit menu for more on editing.

Note: Please be warned that PTVIEW doesn't undo.
Note: It is the best to convert all tab characters to spaces before editing; otherwise, you will see funny characters in place of the tab characters.
Note: You can disallow editing of your file by putting

```
AllowEdit=F
```

in the [GENERAL] section of your INI file. The flag takes effect everytime a new window is opened.

Do you know that...

- Do you know that you can click on the place of interest on the pictures of this help file to find out more information?
- Do you know that you can preprocess a header file by first preprocessing the top level file which inclues it, then dive into it?
- Do you know that you can quickly find out the definition of a symbol by double-clicking it while holding down the *Ctrl* key? ...
- Do you know that you can drag a file from the the File Manager and drop it in PTVIEW to have it opened?
- Do you know that you can drag the "file button" on the File List Bar to another view window (MDI child window) to have the file it represents duplicated?
- Do you know that when you are browsing a symbol (with menu item **Processing | Browse Symbols**) you are presented with the path of file inclusions that leads to the definition of the symbol? The path is shown as a list of file names like:

  <line #> file path

  with the "most recent" (closest to the definition) file comes first. And you can double-click on any one of them to open/dive into the file. ...
- Do you know that PTVIEW uses a hash table to store symbols, and you have control over the size of the hash table? To do this, you need to add the line

  SymbolHashTableSize=*n*

  where *n* is the size of the hash table, in the [GENERAL] section of your INI file. ...
- Do you know that you can have PTVIEW opens a new view window for every file you open with the menu command **File | Open**? To do this, add the line

  OpenWithNewWindow=T

  in the [GENERAL] section of your INI file.
- Do you know that you can increase PTVIEW's preprocess performance by freeing up memory used? For example, symbol list occupies a big chunck of memory; so consider putting the one of the following lines

  DeleteSymbolListBeforeProcess=T

  or

  DeleteSymbolListAfterProcess=T

  in the [GENERAL] section of your INI file.
- Do you know that you can press the *Esc* key to put foreground preprocessing process on the background.