Zeus for Windows Editor

Copyright (C) Jussi Jumppanen 1991-1997
All rights reserved

The Zeus for Windows programmer's editor has been specifically designed for programmers working in the Windows 3.x, Windows 95 and Windows NT environments. This help file provides a complete description of the many features offered by this software. For more information choose from the list of topics shown below:

Contact Details

You can send your mail to the following address.

Jussi Jumppanen

Xidicone Pty Ltd

PO Box 697

Lanecove

NSW 1595

Australia

or you can send your electronic mail (e-mail) to the following address:

Internet: **xidicone@iname.com**

CompuServe: **INTERNET:xidicone@iname.com**

Technical Support, Bugs and General Enquires

If you have any type of problem, suggestions or question regarding this software please contact the author of this software. Please **DO NOT** contact the supplier of the registered software, as in all cases they only process software registrations and will not be able to answer your questions.

If you think you have found a bug with this software then on your bug report please supply a short description of the problem and the version of the software that you are using. The description supplied should also include the steps needed to duplicate the fault. In summary the type of information that should be included should include:

1. **Product Version**

The version number of your product as displayed in the Help | About dialog.

2. **Error Messages**

The exact wording of any error messages that may have been displayed.

3. **Description of Task**

A brief but precise description of the exact task you where attempting. This is most helpful if the problem can be reproduced by following the steps outlined in the problem description.

If possible you also supply a contact address or E-mail address in case further information regarding the problem is required.

If you have any suggestions regarding this product, be they good or bad, then I will be more than interested in hearing of your ideas. If you like the product and you like using it, please let me know or if you hate it, also let me know.

As another source of possible information you may also want to take a look at the Zeus home page which is located at:

http://ourworld.compuserve.com/homepages/jussi

Zeus for Windows Order Form

You can print this order form by selecting the Print Topic option from the File pull-down menu but make sure the printer is setup for portrait mode before you start the printout.

Please note that the Zeus editor is available for a variety of operating systems, so take particular care when ordering the software to insure you get the correct software for your operating system. Also note that the price of the Zeus editor is identical for all operating systems.

Before you place your order you should also check the Zeus home as it will always contain the latest version of the Zeus for Windows editor:

http://ourworld.compuserve.com/homepages/jussi/

When placing your order specify one of the following product names:

Product: **Zeus for Windows** (for Windows 3.x, WfW)

or Product: **Zeus for WIN32** (for Windows 95 or Windows NT)

****

Postage Details

Select the Product: **'Zeus for Windows'** ____ OR **'Zeus for WIN32'** ____

Please supply me the quantity _____ of charged at $US 95.00 each which

includes postage and handling. Total Payment: $US _____

NOTE: All prices quoted are in US dollars only.

Name: _____ Date: _____
Company: _____
Address: _____
_____
_____
City, State: _____ Zip: _____
Country: _____

****

Credit Card Details (if applicable):

Card holders Name: _____
Card Type: MasterCard ___ Visa ___

Card Number: |__|__|__|__| |__|__|__|__| |__|__|__|__| |__|__|__|__|

Card Valid: ____ / ____ to ____ / ____

Phone Number: (____)_____

Signature: _____

****

Order by Mail

If you have a current Visa or MasterCard, you can register this software by just filling in the order form and sending it to:

Xidicone Pty Ltd
PO BOX 697
Lanecove NSW 1595
Australia

Alternatively, you can also register using an international money order, making it payable to Xidicone Pty Ltd. Once again make sure you send a completed order form along with the money order.

Ordering by Internet

You can register Zeus using the online directly from the Zeus home page. To do so just point your browser to the following page:

http://ourworld.compuserve.com/homepages/jussi

You can register Zeus using the online registration services provided by RegNet by just loading one of the following web pages:

http://www.swregnet.com/248p.htm (for Zeus for Windows 3.x, WfW only)

http://www.swregnet.com/249p.htm (for Zeus for Windows 95, Windows NT only)

Alternatively just visit the following RegNet home page and search for the Zeus keyword:

http://www.xmission.com/~wintrnx/regnet

You can register Zeus using the online registration services provided by Albert's Ambry online shareware super store. Just point your web browser to the following page and do a search for the Zeus keyword:

http://www.alberts.com

If you have a valid Visa or MasterCard you can also register this software by filling in the order form and sending it to:

xidicone@iname.com

For those needing to purchase multiple licences of the software, in order to make the task as easy as possible, it is suggested that you use the services provided by the ***RegNet*** system or by ordering the software directly from Xidicone Pty Ltd. If you need help with ordering just send all your questions to **xidicone@iname.com** e-mail address.

Ordering by CompuServe

You can register Zeus via CompuServe by just typing **'GO SWREG'** and following the instructions from there.

The software Registration ID for Zeus for Windows 3.x, WfW is **7380.**

The software Registration ID for Zeus for Windows 95, Windows NT is **8216.**

If you require more help you can send a note to **INTERNET:xidicone@iname.com**

Ordering by Fax

It is also possible to register Zeus by sending a fax copy of the order form. To do so just send an e-mail to the author contact and request the fax number to be used.

Ordering by Purchase Order

Purchase orders (net 30 days) will only be accepted for the purchase of 4 or more copies of Zeus and will only be accepted from government institutions, accredited educational institutions and major corporations. They must be submitted on a purchase order form with a purchase order number and sent to Xidicone Pty Ltd.

Due to the extra work involved in processing purchase orders you are encouraged to use a the online registration services of CompuServe or the other internet registration services.

If you have any questions send all enquiries to Jussi Jumppanen.

Software Registration is Easy!

The Zeus editor is available for the Windows 3.x, Windows 95 and Windows NT operating systems so please make sure you specify the product that you require when placing your order. For more details regarding the different products on offer refer to the order form and for information regarding the latest version of the Zeus editor just visit the Zeus home page.

The current price list (including the site licence prices) for the Zeus editor are shown in the table below.

Number of Copies	Unit Cost ($US)
1-9	$95-00
10-49	$85-00
50-99	$70-00
100-499	$55-00

To order Zeus you may choose from one of the following methods:

Ordering by Mail

Ordering by Internet

Ordering by CompuServe

Ordering by Fax

Ordering by Purchase Order

IMPORTANT NOTE

For any general enquiries, site licences queries, problem reports or complaints regarding this software, please contact the author of the software of the software and not the supplier of the registered software. In all cases the software suppliers only handle registrations an ***do not provide*** any support for this product.

Reason to Register this Software

Once you have registered, you will receive the latest version of the software. It will be free of all reminder messages and the start up CRC checking found in the shareware version. Also limitations placed on some aspect of the shareware version will have been removed. As the software will be the latest version of the product it will also include all the latest bug fixes and any software enhancements.

Your registration also includes free software support by E-mail or mail. Your registration does not include shipping of future versions of the product but you will be placed on the registration list and sent notifications of updates including the list of changes made and the different ways to get the update.

Copyright Notice and Licence Agreement

Licence Agreement

Using this software implies your acceptance of the following terms and conditions:

1. Unregistered software*

The unregistered software may be freely used on any number of machines for any time period. It can be freely copied and distributed on the condition that the distribution will be complete and with no modification to the original software.

2. Registered software*

The registered software may be installed on several machines, with the limitation that the number of simultaneous users will not exceed the number of licences purchased. The distribution of the registered software to third parties is prohibited.

3. Registered or unregistered software

The software disassembly is prohibited. The modification of the dialogs of the registered software with a resource editor is permitted on a personal basis. The distribution of the modified software to third parties is prohibited.

* : The software is unregistered when the licence has not (yet) been purchased. The software is registered once the licence has been purchased.

Licence Warranty

THIS SOFTWARE AND MANUAL ARE PROVIDED "AS IS" AND WITHOUT WARRANTIES AS TO PERFORMANCE OF MERCHANT ABILITY OR ANY OTHER WARRANTIES WHETHER EXPRESSED OR IMPLIED. BECAUSE OF THE VARIOUS HARDWARE AND SOFTWARE ENVIRONMENTS INTO WHICH THIS PROGRAM MAY BE PUT, NO WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE IS OFFERED. THE USER MUST ASSUME THE ENTIRE RISK OF USING THE PROGRAM. ANY LIABILITY OF THE SELLER WILL BE LIMITED EXCLUSIVELY TO PRODUCT REPLACEMENT OR REFUND OF PURCHASE PRICE.

Spectrum Analyser for Windows Version 1.0

The Spectrum Analyser for Windows converts your PC into a full functioning real time FFT Spectrum Analyser. It was designed to convert a standard IBM PC into a cheap but effect piece of electronic test equipment, perfect for the spectrum analysis electronic signals.

The unit offers a 2 channel 8 bit resolution FFT Spectrum Analyser with a bandwidth of 20 Hz to 100 kHz with 2 times over sampling. It performs up to an 8192 point FFT calculation, with true linear correlation as well as supporting the calculation of the probability density function. The hardware is controlled by an MDI Windows application that supports real time graphing of all the results.

For more information visit the following web page:

http://ourworld.compuserve.com/homepages/jussi/sMain.htm

NOTE: As the Spectrum Analyser for Windows consists of a hardware and software component it is only suitable for people with a good knowledge of electronics.

Digital Logic Analyser for Windows Version 1.2

The Digital Logic Analyser for Windows converts your PC into a full functioning, high speed, hardware logic analyser. It was designed to convert a standard IBM PC into a cheap but effect piece of electronic test equipment, perfect for analysing and debugging digital electronic circuits.

The system offers a maximum internal sample rate of 6.00 MHz and an external clock sample rate of about 10 MHz. The unit offers 8 digital input channels, a fully programmable trigger on any combination of 4 input channels and over voltage protection on all inputs channels. The over voltage protection makes it suitable for testing higher voltage digital signals including RS232.

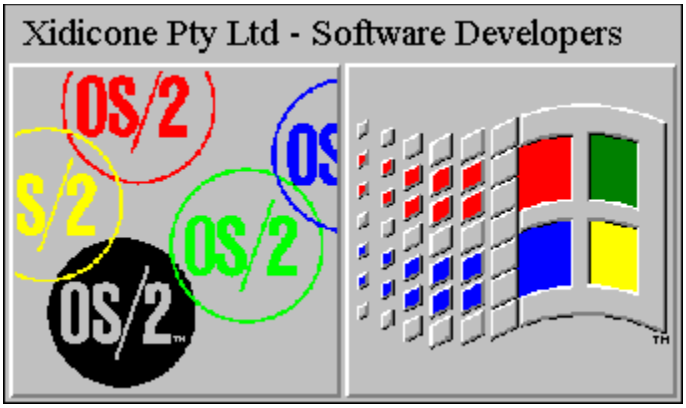
The Digital Logic Analyser is specifically designed for anyone that requires a cheap but effect hardware logic analyser. All this functionality is provided in an easy to use Windows software, which also offers a sophisticated graphical signal tracing display of the sample data taken.

For more information visit the following web page:

http://ourworld.compuserve.com/homepages/jussi/dMain.htm

NOTE: As the Digital Logic Analyser for Windows consists of a hardware and software component it is only suitable for people with a good knowledge of electronics.

About Xidicone Pty Ltd



Quality by design not by chance.

Specialists in OOD, GUI and Client Server software development.

Experts in OS/2 1.x, OS/2 2.x, Windows 3.x and Win 32 software development.

No project too small, no project too large.

Quality software development at a sensible price.

For further information contact Xidicone Pty Ltd.

Xidicone Pty Ltd ACN: 059 825 320

Postal Address:
PO BOX 697
Lanecove NSW 1595
Australia

Voice and E-mail details:
Internet: xidicone@iname.com
CompuServe: INTERNET:xidicone@iname.com
Home Page: http://ourworld.compuserve.com/homepages/jussi/xMain.htm

Registered Trademarks

IBM is a registered trademark of International Business Machines Corporation.

OS/2 is a registered trademark of International Business Machines Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

MS-DOS is a registered trademark of Microsoft Corporation.

Windows is a registered trademark of Microsoft Corporation.

Borland is a registered trademark of Borland International.

zApp is a registered trademark of Inmark.

OK

Close the dialog box saving any changes made.

Update

Close the dialog box saving any changes made.

Close

Close the dialog box.

Help

Provides help on how to use this dialog box.

User Manual

The Zeus for Windows text editor is designed for programmers working on the Windows 3.x, Windows 95 or Windows NT operating systems. To get the best out of this software will require that you the user take some time to learn how to use the many powerful features on offer. To help in this learning process a comprehensive (but not always complete) online help facility has been provided so please take some time to study it. One important point to note is the Zeus editor does assume the install directory has been added to the system path. If you have any problems with any of the features of this editor the first thing you should check is that this is indeed the case. To find out more about the Zeus editor just select from any of the general help topics listed below.

If after studying the online help you still need some information regarding Zeus then remember you can also ask for help using the technical support facility provided.

Command Line Arguments

The Zeus editor can be run from the command line and as well as supporting the standard filename and wildcard file names command line options, Zeus also supports the following command line arguments:

-i INI Filename

This option will configure Zeus using the INI filename specified in the command line rather than using the default INI file (which is located in the Zeus install directory). This option for example could be used to configuration Zeus using an INI file located on a network server and thus the same INI file could be common to several users.

-l LineNumber

This option will start the editing all the files specified in the command line at the given line number position.

-n

Start the Zeus editor without loading the startup history files.

-s

This option will always start a second copy of the Zeus editor.

-u User Path Name

This option allows the user path to be specified using the command line. The user path defines the root directory in which Zeus will search when looking for the configuration files. By default the user path is the just the Zeus installation directory. The Zeus editor uses the following sub directories to load and save configuration information and by default these are all assumed to be located relative to the user path:

zBackup

zConfig

zExample

zHelp

zMacros

zScript

zSpell

This option for example could be used to define the location of the Zeus configuration files and means that these details could be located on a network server and could be common to several users.

Along with these command line arguments the Zeus networking features can also be configured using the Zeus INI file settings.

Network Features

Several directory locations and the configuration files that these directories contain control the configuration of the Zeus editor. The first of these directories is the *user directory,* which by default is the assumed to be the Zeus installation directory. The user directory is used to define the root directory for several other Zeus configuration sub directories. A list of these directories and their purpose is given below:

zBackup

Directory to which the backups are written.

zConfig

Directory containing Zeus configuration information.

zExample

Directory containing useful example information.

zHelp

Directory containing the Zeus Quick Help index files.

zMacros

Directory containing the Zeus keyboard macro files.

zScript

Directory containing the Zeus script macro files.

zSpell

Directory containing the Zeus spell checking dictionaries.

It is possible to change the user directory or any or all of these sub directory locations by just adding the required entry into the Zeus INI file. The names of the INI file entry keys are listed below:

[Network]

UserDirectory=""

Backup=""

Config=""

Example=""

Help=""

Macros=""

Script=""

Spell=""

Make sure that when changing the Zeus INI values that Zeus is not running. This also means that you cannot use Zeus to modify the Zeus INI file values. Also note that the *user directory* can also be configured by using the command line arguments.

As a typical example, consider the case where the user directory is located on a network server and the backup directory is located on the local machine. This could be configured using a set of INI values similar to those shown below:

```
[Network]
UserDirectory="s:\devel\config"
Backup="c:\devel\backup"
```

In addition to these network features several other Zeus features can also be configured via the Zeus INI file settings.

Dialog Boxes

Below is a list of all the options dialog boxes. These allow you to configure different aspects of the behaviour of Zeus editor.

Color Options

Compiler Options

Edit Options

Extension Options

Filter Options

Font Options

Keyboard Options

Macro Options

Project Options

Spelling Options

Template Options

Tags Options

Tool Options

Below is a list of all the general purpose dialog boxes. These dialogs will be used during the course of a typical edit session with Zeus.

DOS Command Line

Edit Extension

Edit Keymap

Edit Template

Execute Macro Script

Find

Goto Bookmark

Goto Line

Load Macro

New Extension

New Keymap

New Template

Open File

Open Project

Print

Print Setup

Program Arguments

Quick Configuration

Quick Help

Quick Search

Repeated Playback

Replace

Save As File

Save Macro

Spelling

Using the Online Help

If you are not familiar with the Windows online help you should first take some time to learn how to use the help system.

As an aid to simplifying the learning process Zeus provides full context sensitive help. What this means is that if you ever have any problems the first thing you should do is hit the F1 key, which will result in help being displayed. The context help provide by Zeus is broken down into three different types of help. This help consists of help on menu commands, help for the active dialog box and help with the keyboard bindings.

To get help on a particular menu command just highlight the menu using the keyboard or mouse and with the menu still highlighted hit the F1 key. The help displayed will be topic specific and will describe the functionality of the menu command in question.

To get help on a particular dialog just activate the dialog and hit the help button or the F1 key. The help system will display help text describing how to use the dialog box and all of the elements of the dialog.

To get help on a keyboard keystroke just select the Help On Keys menu item from the Help menu pull down.

As a final point, when things don't quite go as expected, Zeus will use the status line or a message box's to try to report some type of meaningful error messages. Take the time to read any messages produced, as it may give some insight into why a particular command failed to operate as expected. Please also note these messages and add them to your problem report when it comes time to ask for technical help.

Some Helpful Hints

Add Zeus to the PATH

Zeus expects the install directory to be in the PATH statement of the autoexec file. Please make sure that this is the case as it may well solve any strange bugs that you may be experiencing.

Try the Quick Help Feature

Don't forget that Zeus can quickly search all your programming reference online help (WinHelp.exe) files. To start searching just use the Quick Help Configuration option to install your reference help files.

Improved Keyboard Response

To improve the keyboard responsiveness use the Control Panel Keyboard icon from within the Program Manager (or Start button) to set the repeat rate to the fastest possible setting.

Using the Strip Utility

Some compilers generate listing outputs with 'imbedded error' information while the Zeus error handler would rather prefer an 'error only' listing file. To overcome this problem you can run STRIP.EXE to clean the listing. Refer to the CC.BAT file located in the installation directory for an example of how to use this utility.

Using File Manager Drag and Drop

Zeus supports full file drag and drop. This means files can be automatically loaded from programs like File Manager or Explorer by just double clicking on the file in question or selecting the file(s) and dragging and dropping the selected files onto the running Zeus application.

Opening Program Include Files

To open an include file just place the cursor on the line that contains the include file name and typing the Ctrl-E keystroke (assumes BriefEx keyboard). Zeus will then scan the line for the first valid file name and then attempt to open the file. Zeus will try to locate the file in the current working directory, the directory to which the current document belongs and finally it will attempt to locate the file by using the search rules specified in the Editor Options dialog box.

Opening Files from the Output Window

You can easily open a file that is referenced in the standard output of any utility. For example by running the GREP utility using the Tools Command menu option, it is possible to open the files found by just selecting the line in the output window using a mouse double click or the enter key. This feature can be used with any tool that allows its standard output or standard error to be captured.

Know Bugs and Software Limitations

All effort has been made to provide a product with as few as possible defects. Having said this there will no doubt be several bugs that have yet been located. Also several know bugs have already been identified with this version of the product and these are listed below. In terms of limitations there is no functionality in the release version that is not present in the shareware version. Having said this it is also true that in some cases the functionality although present in the shareware version is a limited version of that found in the actual release version of the product.

Screen Paints with Rubbish

Zeus for Windows uses the Windows palette GDI functions to perform the syntax colour highlighting. This can cause problems on video cards and video drivers that do not support these GDI functions (common in the earlier 16 bit SVGA cards). If this problem occurs try using a newer version of the video driver, or try using the generic SVGA driver supplied with Windows. If possible, try using Zeus on a different machine with a different video card to verify that this is indeed the problem. This ***should not be*** a problem on modern video cards.

File Size Limited to 60,000 for Windows 3.x

The Windows 3.x version of Zeus is limited to a file size of 60,000 lines. This problem is related to the current memory manager used and the limitations of the Windows 3.x scroller. Hopefully these limits will be increased in future versions of the software.

Line length is Limited to 1024 Characters

Zeus only supports a maximum line length of 1024 characters. Lines of greater than 1024 characters will be automatically wrapped. Zeus will wrap the line on a word break and will warn you if any line wrapping was required to load the file.

Clipboard Support

The greatest piece of text that can be copied into the clipboard is 64 KBytes, which equates to about 1000 lines of text. If you try to copy more than 64 KBytes of text, Zeus will fail the command and report an error. This is due to a Windows 3.x clipboard limitation.

Compiler Support under OS/2 2.x and OS/2 3.0

The Zeus compiler support is known not to work with the OS/2 2.x WinOS/2 and OS/2 3.0 Warp products. This is because when Zeus calls WinExec() to spawn the compiler and then monitors the program status using the instance handle returned by that call. Under OS/2, Zeus is incorrectly told that the compiler has ended and as such does not report the errors correctly.

Limited Tool, Macro and Template Support

The tool, macro and template functionality is present in the Zeus shareware version but these features have been limited in the number of tools, macros and templates that can be defined. The release version will not have any such limitation.

Text Marking Features

In addition to the standard keyboard column, block and line marking modes, Zeus also offers the following default mouse marking modes.

Column Marking Mode

To mark a column text region using the mouse hold down the left mouse button while moving the mouse cursor.

Line Marking Mode

To mark a line text region using the mouse hold down the Ctrl key while holding down the left mouse button and moving the mouse cursor.

Block Marking Mode

To mark a block text region using the mouse hold down the Shift key while holding down the left mouse button and moving the mouse cursor.

Once a marked area has been selected the region can be shifted left or right using the MarkShiftLeft and MarkShiftRight edit functions or the Tab and TabBack edit functions. To determine which keys these functions are bound refer to the keyboard mapping section of the help file.

Note that it is also possible to re-configure the default mouse text marking modes by changing the Zeus INI file settings.

Power Editing Features

Zeus offers several edit functions that are designed specifically to help with navigation within the current file and between files. Below is a short list of some of the more useful editing functions designed for this task.

FileOpenInLine

FunctionFindAll

FunctionFindNext

FunctionFindPrevious

To uses these functions they first need to be bound to a keyboard key combination. Consult the keyboard map to see how easy it is to bind these functions.

Also note that the FunctionFindxxx group of functions by default search for C / C++ function prototypes. If you are not writing C / C++ code then you will need to add a regular expression string file extension using the Edit Extension dialog.

These are just a few of the special features offered by Zeus. There are a host of other power editing functions provided. To find out more about the functions on offer just take some time to study the complete function list.

UseExplorer

This allows users of Windows 95 and Windows NT to use the explorer open dialog and not the standard file open dialog. Set this value to 0 or 1.

FileTrim

Zeus adds a single blank line to the end of the file in the form of a carriage return line feed. This option will force Zeus to trim the file of this last blank line. Set this value to 0 or 1.

CheckOnReload

This option is related to the FileReloadCurrent keyboard function (see keyboard mappings) which reloads the current file from disk. This option determines whether the user should be warned before and file reload is carried out. Set this value to 0 or 1.

Clock24Hour

This option will display a 24 hour clock in the status line. Set this value to 0 or 1.

MakeSound

This option determines if Zeus makes a sound when displaying error and warning messages. Set this value to 0 or 1.

PrintFilter

This option determines if Zeus should filter the output before sending it to the printer. When enabled Zeus only sends printable characters to the printer. Set this value to 0 or 1

SmallIconCaption

This option will forces Zeus to reduce the size of the caption text when iconized. When enabled this means that when Zeus is iconized only a short title is displayed.

Mouse Marking

This option controls the default mouse marking modes. Zeus has three mouse marking modes being mouse marking (with left mouse button down), mouse marking with shift key down and mouse marking with the Ctrl key down. The marking modes can be set to the following values:

1 Mark in line mode

2 Mark in block mode

3 Mark in column mode

Caret Setting

Uses these settings to set the size of the cursor caret. These values represent the size in pixels of the caret but are limited to a maximum of the font cell size. Thus you should use values of between 1 and 16 for these settings.

Lines

Zeus uses a regular expression to search for the line number in the compiler output file. In the case where Zeus can not detect the line number this option provides a way of configuring and alternative regular expression to search for line numbers.

Errors

Zeus uses a regular expression to search for errors in the compiler output file. In the case where Zeus can not detect the error this option provides a way of configuring and alternative regular expression to search for errors.

Functions

Zeus uses a regular expression to search for functions when navigation and listing the functions in a file. This option allows the user to define a regular expression that is to be used when searching for functions. As an example for pascal programmers you could use the following expression.

```
^procedure +[_a-z0-9]+
```

AutoExtension

This allows users of Windows 95 and Windows NT to use the explorer open dialog automatic extension feature. With this option enabled the explorer file open dialog will automatically append the current file extension to the file name supplied. Set this value to 0 or 1.

DefaultExtension

This option lets you define the default extension used when a new file is created. By default the extension is TXT.

LineWrapDraw

This option determines if Zeus should draw the line wrapping right hand margin. When enabled a line is draw to represent the current line wrapping column. Set this value to 0 or 1.

INI File Settings

Most of the items in the Zeus INI file can be configured using the Zeus option menu items, but for some of more advanced features, a user interface has not yet been developed. Below is a list of these more advanced INI file values.

Be careful when editing the ZEUS.INI file as it contains important Zeus configuration information. Also make sure you don't use Zeus to edit the INI file as your changes will not take effect. This is because Zeus will overwrite your changed INI file when you close Zeus.

[Editor Options]

CheckOnReload

Clock24Hour

FileTrim

MakeSound

PrintFilter

SmallIconCaption

UseExplorer

AutoExtension

DefaultExtension

LineWrapDraw

[RegularExpress]

Lines

Errors

Functions

[Mouse Options]

MouseMark1

MouseMark2

MouseMark3

[Caret Options]

CaretX

CaretY

CaretInsertX

CaretInsertY

In addition to these features there are also several Zeus network features that can be configured using the Zeus.INI file.

File Navigation

In addition to the advanced file searching available through the use of the output windows Zeus also offers several enhanced file handling techniques to make opening text file as simple as possible. To use these features does require the use of a third party file server application so for example the File Manager, the Explorer or an explorer type open dialog box could be used. Below is a description of the types of additional file handling that is possible.

Drag and Drop Support

It is possible to open one or more files by selecting the files from the file server application, dragging them to a running Zeus application window or its minimised icon and then dropping them. This action will open the selected files for editing.

Open Using File Association

Once Zeus for Windows has an association to a file extension you can open a file by just double clicking on the file within the file server application or by selecting the file and hitting enter key. To associate the file, use the Explorer, the File Manager or the Regedit application.

Command Line Support

For Windows 95 and Windows NT it is possible to run Zeus from the DOS command line. This means that you can also get Zeus to load files using the command line. For example to load all the 'cpp' files in the current directory you could use the following command from any DOS command line.

```
Zeus *.cpp
```

For Windows 3.x users this is not a valid option but Windows 3.x users can simulate this feature using the File Manager File | Run command line instead.

DDE Open Command Support

Zeus supports the DDE open command. This command has the following format:

```
"[open("%1")]"
```

where %1 is the full path name of the file to be opened. You can also use the ShellExecute() Windows API to open files using Zeus, provided you first associate the file extension with the Zeus application.

Starting a Single Instance Of Zeus

The Zeus editor is designed to only run a single instance of the application almost all the time. This means that if you try to open a second instance of Zeus the first instance will be activated and will display the file that was to be displayed by the second instance and finally the second instance of Zeus will terminate.

The one exception to this behaviour is when you specify a wild card as part of the argument list to the second instance. For example if you start the second instance of Zeus using the following command line:

```
Zeus *.c
```

Then the second instance will start independently of the first. In this case you will have two instances of Zeus running. The same result can also be achieved by using the -s command line argument. For example the following command line will always start a new copy of the Zeus editor:

```
Zeus -s Test.cpp
```

Possible Problem with GPF's

This design feature does introduce a possible problem. If for some unexpected reason the first instance of Zeus should General Protection Fault (GPF) Zeus will not have done the proper clean up on exit. This means that the next instance of Zeus will still think the first instance is running and so it will also terminate. Under this circumstance start a second version of Zeus with a wild card in the argument list. This will force the second instance of Zeus to start. If you then exit this newly run version of Zeus the system will be returned to its proper state and functionality will have returned to that prior to the GPF.

Command Line TAG Values

Zeus offers pre-defined tags that can be used as arguments for any of the tools or directly from the DOS command line dialog box. You can also use the tags to configure the project and compiler command lines and they can also be used when writing or running macro scripts.

The specified tag is expanded to the appropriate value using the details of the currently active session. In most cases the tag information is derived from the currently active document. For the case where the tag does refer to a particular document detail yet the currently active window is not document related, the tag gets interpreted using the following rules.

1) For the case where no window is active the tag will remain unexpanded.

2) For the case where the current window is a compiler/project/output window the tag will try to extract the details required from the currently selected line.

3) Failing this the name of the data file currently being displayed in the compiler/project/output window will be used as the source of the document information.

If an unknown macro tag is used then it will be left unexpanded. Also note that in some cases the macro will get expanded to a null string. For example if you used the $FD tag while the active window was a 'untitled.txt' new file then the result would be a null string as the current document does not contain any file directory information in its name.

Below are listed of all the TAGS supported and a description of its purpose.

Long Name	Short Name	Description
$File	$F	File name including extension
$Ext	$E	File extension (includes '.')
$ExtBase	$EB	File extension (excludes '.')
$FileBase	$FB	File base name excluding extension
$FileName	$FN	Fully qualified file name
$FileDir	$FD	File directory (includes '\')
$FileDirBase	$FDB	File directory base (excludes '\')
$FileDrive	$FDR	Drive letter (includes '\')
$FileDriveDir	$FDD	Drive letter and file directory (includes '\')
$Line	$L	Current line number
$CurrentDir	$CD	Current working directory
$Column	$C	Current column number
$Marked	$M	Marked text (maximum of one line only)
$Word	$W	Current word excluding quoted text
$WordEx	$WEX	Current word or quoted text
$ProjectDir	$PD	Project directory (includes '\')
$ProjectFile	$PF	Project file name
$ProjectBase	$PB	Project file base name excluding extension
$ProjectDrive	$PDR	Project drive (includes '\')
$SearchPath	$SP	Search path as defined in option editor dialog.
$SystemDir	$SD	Windows system directory (excludes '\')
$WindowsDir	$WD	Windows directory (excludes '\')
$ZeusDir	$ZD	Zeus binary directory (includes '\')
$$	$$	Insert a single $ character

Understanding Project Support

Zeus by default captures the standard output generated by the project build process.One possible problem is that the builder you are running is in fact writing to standard error, so the first thing to check is try running the build with the Capture standard error option set.

One other important point to note is that the Zeus editor assumes the install directory has been added

to the system path. If you have any problems with any the project support the first thing ***you should check*** is that this is indeed the case. Also make sure the builder EXE file you are trying to use is also included in the system path.

If you are still experience problems making the project open the Project Options dialog box and select the "Help to debug this session" checkbox. This option will add a pause command to the compiler batch file, which gives you a chance to look for possible error messages generated during the compile process. Also make sure the display mode is set to normal or maximised.

If this does not help the next step is to open up DOS command shell (i.e. select the Tools DOS Shell menu item) and go to the same directory as that of the project file you are trying to build. Here you will find a file called 'ZeusCC.BAT' which is a batch file crated by Zeus and this file is run whenever the project build is required. At the DOS command line run this batch file by typing in 'ZeusCC.bat' and again check to see if any error messages are reported. To get a feel for what is going on, also take a look at the 'ZeusCC.BAT' batch file using an editor and you may want to take out the file redirection commands contained in the batch file.

If this does not help, try running the Project Options project build command line directly from the DOS command line. If the command entered into the Project Options dialog does not run from the DOS prompt it will definitely not work when run from within the Zeus Editor. Correspondingly Zeus should have no trouble running any project build command that also runs when used from a DOS command line prompt.

For those not interested in using the make utility to build their project it is also possible to use a simple batch file to build the project. Any example of this type of project file is contained in the "zExample\pm.bat" which is located in the installation directory.

Understanding Compiler Support

Zeus by default captures the standard output generated by the compiler. One possible problem is that the compiler you are running is in fact writing to standard error, so the first thing to check is try running the compile with the Capture standard error option set.

One other important point to note is that the Zeus editor assumes the install directory has been added to the system path. If you have any problems with any the compiler support the first thing ***you should check*** is that this is indeed the case. Also make sure the compiler EXE file you are trying to use is also included in the system path.

If you are still experience problems compiling a file open the Compiler Options dialog box and select the "Help to debug this session" checkbox. This option will add a pause command to the compiler batch file, which gives you a chance to look for possible error messages generated during the compile process. Also make sure the display mode is set to normal or maximised.

If this does not help the next step is to open up DOS command shell (i.e. select the Tools DOS Shell menu item) and go to the same directory as that of the file you are trying to compile. Here you will find a file called 'ZeusCC.BAT' which is a batch file crated by Zeus and this file is run whenever the file is compiled. At the DOS command line run this batch file by typing in 'ZeusCC.bat' and again check to see if any error messages are reported. To get a feel for what is going on also take a look at the 'ZeusCC.BAT' batch file using an editor and you may want to take out the file redirection commands contained in the batch file.

If this does not help try running the Compiler Options compile command line directly from the DOS command line. If the command entered into the Compiler Options dialog does not run from a DOS prompt it will definitely not work when run from within the Zeus Editor. Correspondingly Zeus should have no trouble running any compilation command that also runs when used from a DOS command line prompt.

Compilers With Strange Error Messages

Some compilers produces a very much ***non standard*** types of error message in that they give no indication that the line of output produced is in fact an error or a warning. This is known to be true for the **DJGPP, ADA** and **Java** compilers and there are undoubtedly more. For these compilers you will need to add the following line to the Zeus INI file:

```
; Add this for error handling in DJGPP and ADA error messages
[RegularExpress]
Errors="[^ \t\")(<'`]*[a-zA-Z0-9_]*\\.[a-zA-Z0-9_]*[^ :\t\")(>'`*}]"
```

or this:

```
; Add this if you require support for JAVA error messages
[RegularExpress]
Errors="[:][0-9]+[:]"
```

These are not ideal solutions as they can generate false error reporting but until the compilers writers make life a little easier by giving some indication of an error in their output listings these are a good work around solutions. Make sure you use another editor to edit the Zeus INI file (like **notepad.exe)** and make sure Zeus is not already running when you edit the file.

Compilers that write to Standard Error

Some compilers write their error output to standard error and not standard output. For these compilers make sure that you check the "Capture standard error" option. For this feature to work the Zeus install directory ***must be*** in the system **PATH** statement.

The **DJGPP** compiler is one such compiler but with this compiler you can also achieve the same result by adding the following line to the DJGPP environment variables contained in the 'SETDJ387.BAT' file:

```
set GO32=2r1
```

DOS 125 Character Line Limit

One common problem is that the DOS command line is limited to some 125 characters. This limit is quickly reached when running a compile with lots of compile options. To get around this problem use a response or configuration file in the command line and place the compile options inside the response file.

Here is a configuration file example for the Borland C++ compiler:

C:\BCC\BIN\BC +C:\ZEUS\ZEUS.CFG SAMPLE.CPP

where the contents of ZEUS.CFG response file could be:

/Ic:\bc4\include;c:\bc4\owl\include -c -w;

Here is an response file example for the Microsoft C++ compiler:

C:\MSVC\BIN\CL @C:\ZEUS\ZEUS.CFG SAMPLE.CPP

where the contents of ZEUS.CFG response file could be:

/c /Od /AL /W3

Fully Qualify the Response File

It is always best to fully qualify the response file name. By using a fully qualified path for the configuration and response files (see the examples given above) the compiler will always locate the response file. If you don't fully qualify the name of the response file the compiler will only look for the file in the current directory. This means that for the compile to work correctly the response file must be located in the same directory as the text file being compiled.

Don't Forget the Environment Variables

Some command line compilers require the DOS environment variables to be set correctly. For example the Microsoft compile use the LIB and INCLUDE environment variables. For more information on the typical compiler switch settings, the use of environment variables and the use of command line response files, refer to the documentation supplied with the compiler.

Using the Output Windows

Zeus captures the output of tools, the DOS command line, compilers and project make utilities in special output windows. In some instances the standard output, compiler output or project output windows may display no output when it is expected that there should be output available. In this case it is possible that the spawning process failed to work correctly. This can be due to the program not being in the path, the program name being spelt incorrectly or the program actually not running correctly. In these cases check that the tool, compiler or project make is setup correctly. For more information on the different types of output windows click on one of the items listed below.

Standard Output

Compiler Output

Project Output

Understanding Tool Support

Zeus allows you to use third party tools to further enhance the functionality of Zeus. To get help on setting up these tools refer to the Tool Options dialog. This section describes how the internals of this feature works by providing example code for two tools provided. This should enable you to easily write your own tools or enhance the tools provided. These examples can be run from the Tools menu and the source code is located in the zExample sub directory.

CTAG.EXE Example

The CTAG.EXE is a quick port of the CTAG.C file that comes with the ELVIS public domain editor (refer to the CTAG.TXT for more information). The CTAG program is designed to scan a C/C++ file(s) for static definitions, global definitions and function's, writing the results of the scan to standard output.

To see how it all works open any C or C++ file, for example you open the CINC.CPP file located in the Zeus zExample directory. Then with the CINC.CPP as the active document, run the CTAG.EXE by selecting the Tools | C/C++ Tag Information menu option. Alternatively you could run the CTAG.EXE using the command line features provided by Zeus by typing in the following commands.

1) Select the Tools DOS Command Line menu option
2) Enter the following command line and run the command CTAG.EXE *.c

To highlight the power of this tool select one of the lines of output produced and hit the enter key or use the mouse double click. If all went well the selected file should now be the active file and it will have been automatically loaded for editing and the cursor should now be located at the line referenced by the output.

For more help on how to use the CTAG.EXE program enter the following DOS command line:

```
CTAG.EXE
```

If you require a more functional implementation of the CTAGS program then the "Exuberant CTAGS" comes highly recommended. This program is written by Darren Hiebert <darren@hiebert.com> and is available from the following web page:

```
http://darren.hiebert.com
```

To use "Exuberant CTAGS" within the Zeus environment just add the program to the Zeus Tools menu using the following command line:

```
CTAGS.EXE   -x   $FN
```

CINC.EXE Example

The CINC.EXE is a simple tool that scans the file supplied and builds up an include file tree structure. This tool was designed for C/C++ files but it could easily be modified to support other types of programming languages. The output produced by CINC can be displayed as a formatted or just in a simple line structure. The tools can also scan system include files if requested to do so. It assumes a system file is included using the #include <> format.

To see how it works open any C or C++ file, for example you could open the zExample\CINC.CPP file. Then select the Tools | C/C++ Include Details menu option to run the tool.

Alternatively you could run the CINC.EXE using the command line features provided by Zeus. To do so type in the following command:

1) Select the Tools DOS Command Line menu option
2) Enter the following command line and run the command

   ```
   CINC.EXE   -f -s zExample\CINC.CPP
   ```

To highlight the power of this tool select one of the lines of output produced by this tool and hit the enter key or use the mouse double click. If all went well the selected file should now be the active file and it will have been automatically loaded for editing and the cursor should now be located at the line referenced by the output.

For more help on how to use the CINC.EXE program enter the following DOS command line

CINC.EXE

Standard Output

The standard output window is used to capture the standard output of any of the tools that are run or for capturing the output of any of the DOS command line functions. Note that in these cases Zeus by default will capture the standard output produced. In some cases the output being produced may in fact be going to standard error in which case it will not be captured. To capture the standard error produced by the tools make sure the Capture standard error option is set.

It is possible to quickly load a file from the standard output window by selecting a line from the standard output and hitting the Enter key or using the mouse double click command. This feature only works if the line selected contains a valid file name.

DIR Example

To list all the files in the current directory you would do the following.

Tools Menu, DOS Command Line DIR *.*

GREP Example

Assuming you have a GREP tool you could GREP the zExample directory for the word BOOL: using the following command.

Tools Menu, DOS Command Line GREP BOOL \zeus\zExample*.cpp

This example also assumes that you have a GREP.EXE in the current PATH statement.

CTAG Example

Open a C file like the zExample\CTAG.CPP and with this as the active document window run the following command from the Tools pull down menu.

Tools Menu, C Include or Tools or the C Tags tools

Project Output

The project output window captures the output of the project build process. It will display the errors and warnings generated by building the currently open project. This window will contain no output if no project is currently loaded. It is possible to quickly load a file from the project output window by selecting a line from the project output and pressing the Enter key or using the mouse double click command. This feature only works if the line selected contains a valid file name. If you are having trouble running the project make refer to the section on setting up the project.

Compiler Output

The compiler output window captures the output of the compile process. It will display the errors and warnings generated by compiling the currently active document. It is possible to goto to the line number of the compile error using the enter key or by using the next and previous error commands. If you are having trouble running the compiler refer to the section on setting up the compiler .

Search Path Logic

The Zeus file handling is built around a complex file searching algorithm. Zeus provides several methods by which a user can indirectly initiate a file search and load operation but depending on the situation in question a different search and load method will be used.

For example, when the a search and load is attempted on a file that is related to the document currently being displayed (i.e. an attempt to open an include file from within the current document) then Zeus first checks for the file in the same directory as that of the file currently being displayed. If this fails to locate the file Zeus will then search for the file using the search path as defined in the Editor Options dialog and finally it will also search the INCLUDE environment variable.

As another example if a search an load is attempted from the captured output produced by of a particular tool then working directory of the tool will be first checked. If this fails then a similar search to that described in the first example will be attempted.

One exception to both these case is the case where the name of the file being loaded is fully qualified. In this case the file is not search for as it can be loaded directly using the qualified name.

Open File Dialog

The Open File dialog allows you to open an existing document for editing. For more information on each of the different sections of this dialog click on one of the items listed below.

File Name

List of File Types

Directories

Drives

OK

Cancel

Open Project Dialog

The Open Project dialog allows you to select the project to be opened. The project selected becomes the currently active project, making it available for use by the Project menu commands. For more information on each of the different sections of this dialog click on one of the items listed below.

- File Name
- List of File Types
- Directories
- Drives
- OK
- Cancel

Load Macro Dialog

The Load Macro dialog allows you open a macro file. The macro file selected becomes the currently active macro making it available for use by the Macro menu commands. For more information on each of the different sections of this dialog click on one of the items listed below.

File Name

List of File Types

Directories

Drives

OK

Cancel

User Input Dialog

The User Input dialog allows you to enter macro script specific information. Just fill in the required input fields an hit the OK push button to complete the process. For more information on each of the different sections of this dialog click on one of the items listed below:

OK

Cancel

Save As File Dialog

The Save As File dialog allows you to save the currently active file to a file of a different name. If the file name entered represents a file that already exists you will be asked to confirm the fact that the old file will be over written. For more information on each of the different sections of this dialog click on one of the items listed below.

File Name

List of File Types

Directories

Drives

OK

Cancel

Save Macro Dialog

The Save Macro dialog allows you to save the current macro to file. If the name of the macro file entered represents a macro that already exists you will be asked to confirm the fact that the old macro file will be over written. Once a macro has been saved it can be loaded at any time in the future using the Load Macro dialog box. For more information on each of the different sections of this dialog click on one of the items listed below.

File Name

List of File Types

Directories

Drives

OK

Cancel

File Name

Type or select the name of the file required. This listbox contains all the files with the same extension as select in the List Files Of Type box or the files that match the wild card expression entered. For example, to see a list of files of a particular extension just type an asterisk (*), a period, and the three character extension required.

List Files Of Type

Select the type of file you want to be displayed in the files listbox.

Drives

Select the drive that is to be examined.

Directories

Select the directory that is to be examined.

Read Only

Opens a document for viewing only. You can not save changes to a file you open as read only unless you then save the document with a different filename.

Print Setup Dialog

The Print Setup dialog is used to select a printer, the printer connection and to configure that printer for use. For more information on each of the different sections of this dialog click on one of the items listed below.

Default Printer

Specific Printer

Orientation

Paper

Options

OK

Cancel

Print Dialog

The Print dialog allows you to print from the currently active document. For more information on each of the different sections of this dialog click on one of the items listed below.

- Print Range
- Quality
- Print to File
- Copies
- Setup
- OK
- Cancel

Range

Select the range of the document to be printed. This can be the whole document, the currently marked region or just a selected page range.

Quality

Select the quality of the print out required.

Copies

Select the number of copies of the print out required.

Print To File

Send the output to a file and not to the printer.

Setup

Setup the currently selected printer prior to doing the actual printing.

Specific Printer

Select the printer you want to use. Only printers that have been installed will appear in the list.

Default Printer

Use the default printer for output

Options

Allows you to perform printer specific configuration.

Paper

Select the type of paper being used.

Orientation

Select the required orientation of the print out.

Filter Options Dialog

The Filter Options dialog allows you to define file filter definitions which then get added to the List Files of Type drop down list located in the Open File and Save As File dialog boxes. To define a filter requires you to enter a filter descriptive string into one of the filter edit fields. The data entered ***must be*** in the correct format for these dialogs to operate correctly, so please be careful when entering the data. And example of a suitable filter is shown below.

```
Clipper Files (add your text) |*.prg;*.ch|
```

The filter string is made up of two parts. The first is a descriptive name for the filter and the second is a list of the file extensions associated with the filter description. The special character '|' is used to mark the termination point of the two parts. To add your own filter use the copy to clipboard feature of the online help to make a copy of the example filter and use it as a template for your own filter or use the example button located on this dialog. For more information on each of the different sections of this dialog click on one of the items listed below.

File Extension Filters

Use the Default Filters

Example

Update

Cancel

Help

Warning: The file filter data supplied is not validated in any way so please enter it carefully paying particular attention to the lack of white space in the filter part of the string. If the data is not entered correctly the program operation can not be predicted, so please ensure that the data is entered in a format identical to that shown above!

File Extension Filters

You can enter up to 9 file extension filters by just entering the data into the appropriate edit fields.

Use the Default Filters

If you wish to use the pre-defined default filters (default Zeus filters) just select this option.

Example

The example button will add an example filter to the clipboard. You should use the Ctrl + V command to paste the filter into one of the empty edit field.

Repeat Count

This is the number of time you wish to repeat the macro playback.

Play

This will play back the macro the repeat count number of times.

Repeated Playback Dialog

The Repeated Playback dialog allows you to run the macro a repeated number of times. For more information on each of the different sections of this dialog click on one of the items listed below.

Repeat Count
Play
Cancel
Help

DOS Command Line Dialog

The DOS Command Line dialog allows you to run commands as if you where at a DOS prompt. Any command that you could run at the DOS prompt you can also run from this dialog. For more information on each of the different sections of this dialog click on one of the items listed below.

Directory

Arguments

Run

Cancel

Help

As an example of using the command line enter the following into the arguments section of the dialog and hit the run button.

DIR *.EXE

Remember that the you can also include one of the many Zeus TAG macros in the arguments and directory entry fields.

DOS Directory

This is the directory in which the command is to be run. When left blank the directory defaults to the current directory.

DOS Arguments

These are the arguments to be used. This is effectively the DOS command line.

DOS Run

This button runs the command and captures the output for display. If the result of the command is a blank screen it is possible you entered the command incorrectly or the EXE file does not exist or is not in the search path or the program is writing to standard error.

Program Arguments Dialog

The Program Arguments dialog allows you to supply the information to the program that is about to be run. To define a program refer to the Tools Option dialog section. For more information on each of the different sections of this dialog click on one of the items listed below.

Directory

Arguments

Run

Cancel

Help

Remember that the you can also include one of the many Zeus TAG macros in the arguments and directory entry fields.

Document Command (Spelling menu)

The Document command lets you check the currently active document for spelling errors. To use this option make sure that the spelling engine has been correctly configured using the Spelling Options dialog.

Current Word Command (Spelling menu)

The Current Word command lets you check the current word for spelling errors. To use this option make sure that the spelling engine has been correctly configured using the Spelling Options dialog.

Change Word To

This listbox gives you a selection of possible replacement words to choose from.

Change

The change button allows you to change the incorrectly spelt word using the currently selected alternative word or the modified text in the 'Word not found in dictionary' edit field.

Ignore

This button will ignore the spelling of the word. Please note that once the word has been ignored it is assumed to be spelt correctly for the remainder of the document and all subsequent spell checks of the same document up until the time the document is closed.

Add

The Add button allows you to add words to a custom dictionary. The custom dictionary is used for all spell checks and as such this option effects all future spell checks.

Options

The Options button allows you to configure the spelling engine.

Spelling Dialog

The Spelling dialog allows you to control the spelling engine. The dialog will offer alternative suggestions to words that are not found in the dictionary and also allows you to change or ignore the word that is suspected to be incorrectly spelt. For more information on configuring the spelling engine refer to the Spelling Options dialog section. For more information on each of the different sections of this dialog click on one of the items listed below.

Change word to

Change

Ignore

Add

Options

Cancel

Help

Keyboard Definitions

Zeus offers a highly configurable keyboard handling mechanism, which is based on the standard 101 keyboard layout. The keyboard handler lets you bind almost any key to any one of the listed text editing functions . Also provided are a collection of pre-defined keyboard mapping based on some of the more popular editors. You may choose to uses one of these pre-defined mapping's, modify an existing mapping to suit your requirements or define a totally new keyboard mapping.

For more information on the pre-defined keyboard mapping's provided select an item from the list below:

Brief Version 3.10 Keyboard

Brief Extended Keyboard

Epsilon Keyboard

WordStar Keyboard

Default Keyboard

Important Keyboard Information

The Windows operating system provides quick menu activation by using the Alt+<MenuPrefix> keyboard combination. For example when running the NOTEPAD.EXE it is possible to activate the File menu but entering the Alt+F key combination. This can cause problem for several of the editors keymaps that Zeus emulates as sometimes these keys have been remap to some other editor functionality. To solve this conflict of interest Zeus will always give first preference to the editor keymap and only pass on the key combination to the Windows menu if it is not already in use by the currently active editor keymap. So if you find that the menu accelerator keys are not working then you will need to edit the currently active keyboard mapping and remove or remap the conflicting keyboard entry.

One other solution to this problem is to use the standard Windows two step menu activation procedure as this is always guaranteed to work. For example rather than using the Alt+F menu activation it is also possible to achieve the same result by first pressing the Alt key (and releasing it) then pressing the F key.

As a final note also remember that Zeus binds keys as a single unit. This means that the Alt+W key binding is not the same as the Alt key followed by the W key with a key release in between.

For more information on how to configure the keyboard mapping system refer to the Keyboard Options dialog for more details.

Default Keyboard

The default keyboard mapping provided by Zeus is just a basic keyboard mapping that supports the Windows Common User Access text interface and provides general text cursor navigation. To find out more information about the keystrokes provided select from one of the items listed below:

Keyboard Mapping

Keyboard Function Groups

Default Keyboard Mapping

Below is complete list of keyboard commands provided by the default keyboard mapping. For a list of the more commonly used keystrokes ordered by functional group refer to the keyboard function groups section.

Alt+C	MarkColumnToggle
Alt+D	FileListDisplay
Alt+L	MarkLineToggle
Alt+M	MarkBlockToggle
Alt+Q	HelpQuickHelp
Alt+Z	ToolsShell
Alt+B	WordDeleteNext
Ctrl+A	MarkSelectAll
Ctrl+C	MarkCopyEx
Ctrl+F	SearchWordCurrent
Ctrl+G	LineGoto
Ctrl+H	ReplaceWordCurrent
Ctrl+N	FileNew
Ctrl+O	FileOpen
Ctrl+P	FilePrint
Ctrl+R	MacroRepeat
Ctrl+S	FileSave
Ctrl+V	MarkPasteEx
Ctrl+X	MarkCutEx
Ctrl+Y	Redo
Ctrl+Z	Undo
Ctrl+Backspace	WordDeletePrevious
Ctrl+Delete	MarkDeleteEx
Ctrl+End	MoveLineEnd
Ctrl+F1	HelpQuickHelp
Ctrl+Home	MovePageStart
Ctrl+Insert	MarkCopyEx
Ctrl+Left	MoveWordPrevious
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Return	EnterNext
Ctrl+Right	MoveWordNext
Ctrl+Shift+Left	MoveWordPrevious
Ctrl+Shift+Right	MoveWordNext
Shift+Alt+F1	MacroExecute1
Shift+Alt+F1	MacroExecute1
Shift+Alt+F11	MacroExecute11
Shift+Alt+F12	MacroExecute12
Shift+Alt+F13	MacroExecute13
Shift+Alt+F14	MacroExecute14
Shift+Alt+F15	MacroExecute15
Shift+Alt+F2	MacroExecute2
Shift+Alt+F3	MacroExecute3
Shift+Alt+F4	MacroExecute4
Shift+Alt+F5	MacroExecute5
Shift+Alt+F6	MacroExecute6
Shift+Alt+F7	MacroExecute7
Shift+Alt+F8	MacroExecute8

Shift+Alt+F9	MacroExecute9
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F10	ToolsExecute10
Shift+Ctrl+F11	ToolsExecute11
Shift+Ctrl+F12	ToolsExecute12
Shift+Ctrl+F13	ToolsExecute13
Shift+Ctrl+F14	ToolsExecute14
Shift+Ctrl+F15	ToolsExecute15
Shift+Ctrl+F2	ToolsExecute2
Shift+Ctrl+F3	ToolsExecute3
Shift+Ctrl+F4	ToolsExecute4
Shift+Ctrl+F5	ToolsExecute5
Shift+Ctrl+F6	ToolsExecute6
Shift+Ctrl+F7	ToolsExecute7
Shift+Ctrl+F8	ToolsExecute8
Shift+Ctrl+F9	ToolsExecute9
Shift+Delete	MarkCutEx
Shift+End	MoveLineEnd
Shift+F3	SearchPrevious
Shift+F6	WindowPrevious
Shift+Home	MoveLineHome
Shift+Insert	MarkPasteEx
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Return	EnterOpen
Shift+Right	MoveLineRight
Shift+Tab	TabBack
Backspace	Backspace
Clear	MovePageCenter
Delete	MarkDeleteEx
End	MoveLineEnd
Escape	MarkHide
F1	HelpIndex
F3	SearchNext
F6	WindowNext
F7	MacroRecordToggle
F8	MacroPlay
Home	MoveLineHome
Insert	InsertModeToggle
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
PageDown	MovePageDown
PageUp	MovePageUp
Return	Enter
Right	MoveLineRight
Tab	Tab

Default Keyboard Functions

Below is a list of some of the more common default keyboard commands group by functionality. For a complete list of the default keyboard commands ordered alphabetically refer to the keyboard mapping section.

File Functions

Alt+D	FileListDisplay
Ctrl+N	FileNew
Ctrl+O	FileOpen
Ctrl+P	FilePrint
Ctrl+S	FileSave

Navigation Functions

Shift+End	MoveLineEnd
Shift+Home	MoveLineHome
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Clear	MovePageCenter
Home	MoveLineHome
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
PageDown	MovePageDown
PageUp	MovePageUp
Shift+Right	MoveLineRight
End	MoveLineEnd
Right	MoveLineRight
Ctrl+End	MoveLineEnd
Ctrl+Home	MovePageStart
Ctrl+Left	MoveWordPrevious
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Right	MoveWordNext
Ctrl+Shift+Left	MoveWordPrevious
Ctrl+Shift+Right	MoveWordNext
F6	WindowNext
Shift+F6	WindowPrevious
Ctrl+G	LineGoto

Undo Functions

Ctrl+Y	Redo
Ctrl+Z	Undo

Mark Functions

Alt+C	MarkColumnToggle
Alt+L	MarkLineToggle
Alt+M	MarkBlockToggle
Ctrl+A	MarkSelectAll
Ctrl+C	MarkCopyEx
Ctrl+V	MarkPasteEx

Ctrl+X	MarkCutEx
Ctrl+Delete	MarkDeleteEx
Ctrl+Insert	MarkCopyEx
Shift+Insert	MarkPasteEx
Shift+Delete	MarkCutEx
Delete	MarkDeleteEx
Escape	MarkHide

Line Functions

Backspace	Backspace
Insert	InsertModeToggle
Return	Enter
Ctrl+Return	EnterNext
Shift+Return	EnterOpen
Tab	Tab
Shift+Tab	TabBack
Alt+Backspace	WordDeleteNext
Ctrl+Backspace	WordDeletePrevious

Search Functions

Ctrl+F	SearchWordCurrent
Ctrl+H	ReplaceWordCurrent
F3	SearchNext
Shift+F3	SearchPrevious

Tool Functions

Alt+Z	ToolsShell
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F11	ToolsExecute11
Shift+Ctrl+F12	ToolsExecute12
Shift+Ctrl+F13	ToolsExecute13
Shift+Ctrl+F14	ToolsExecute14
Shift+Ctrl+F15	ToolsExecute15
Shift+Ctrl+F2	ToolsExecute2
Shift+Ctrl+F3	ToolsExecute3
Shift+Ctrl+F4	ToolsExecute4
Shift+Ctrl+F5	ToolsExecute5
Shift+Ctrl+F6	ToolsExecute6
Shift+Ctrl+F7	ToolsExecute7
Shift+Ctrl+F8	ToolsExecute8
Shift+Ctrl+F9	ToolsExecute9

Macro Functions

F7	MacroRecordToggle
F8	MacroPlay
Ctrl+R	MacroRepeat
Shift+Alt+F1	MacroExecute1
Shift+Alt+F1	MacroExecute10
Shift+Alt+F11	MacroExecute11
Shift+Alt+F12	MacroExecute12
Shift+Alt+F13	MacroExecute13
Shift+Alt+F14	MacroExecute14

Shift+Alt+F15	MacroExecute15
Shift+Alt+F2	MacroExecute2
Shift+Alt+F3	MacroExecute3
Shift+Alt+F4	MacroExecute4
Shift+Alt+F5	MacroExecute5
Shift+Alt+F6	MacroExecute6
Shift+Alt+F7	MacroExecute7
Shift+Alt+F8	MacroExecute8
Shift+Alt+F9	MacroExecute9

Help Functions

F1	HelpIndex
Alt+Q	HelpQuickHelp
Alt+F1	HelpQuickSearch
Ctrl+F1	HelpQuickHelp

Brief Version 3.10 Keyboard

The Brief Version 3.10 keyboard mapping can be used to configure Zeus for a Brief keyboard look and feel. To find out more information about the keystrokes provided select from one of the items listed below:

Brief Version 3.10 Keyboard Mapping

Below is list of keyboard commands provided by the Brief 3.10 keyboard mapping. Note that some Brief commands that are not supported may be supported in future versions while others commands are just not applicable as they have little or no meaning in a Windows MDI environment. For a list of the more commonly used keystrokes ordered by group refer to the keyboard function groups section.

Alt+-	WindowPrevious
Alt+0	BookMarkDrop0
rAlt+1	BookMarkDrop1
Alt+2	BookMarkDrop2
Alt+3	BookMarkDrop3
Alt+4	BookMarkDrop4
Alt+5	BookMarkDrop5
Alt+6	BookMarkDrop6
Alt+7	BookMarkDrop7
Alt+8	BookMarkDrop8
Alt+9	BookMarkDrop9
Alt+A	MarkColumnToggle
Alt+B	FileListDisplay
Alt+Backspace	WordDeleteNext
Alt+C	MarkColumnToggle
Alt+D	LineDelete
Alt+E	FileOpen
Alt+End	MoveLineRightEdge
Alt+F	FileName
Alt+F1	HelpQuickSearch
Alt+F10	CompilerCompile
Alt+F5	SearchReverse
Alt+F6	ReplaceReverse
Alt+F7	MacroLoad
Alt+F8	MacroSave
Alt+G	LineGoto
Alt+H	MarkWordCurrent
Alt+Home	MoveLineLeftEdge
Alt+I	InsertModeToggle
Alt+J	BookMarkGoto
Alt+K	LineDeleteEnd
Alt+L	MarkLineToggle
Alt+Left	MoveWordStart
Alt+M	MarkBlockToggle
Alt+Minus	WindowPrevious
Alt+N	WindowNext
Alt+O	FileSaveAs
Alt+P	FilePrint
Alt+Q	HelpQuickHelp
Alt+R	FileInsertAtCursor
Alt+Right	MoveWordEnd
Alt+S	SearchWordCurrent
Alt+T	ReplaceWordCurrent
Alt+U	Undo
Alt+V	Version
Alt+W	FileSave
Alt+X	FileExit
Alt+Z	ToolsShell

Backspace	Backspace
Clear	MovePageCenter
Ctrl+,	FunctionFindPrevious
Ctrl+-	FileClose
Ctrl+.	FunctionFindNext
Ctrl+[	BraceMatch
Ctrl+]	BraceMatch
Ctrl+B	ScrollLinePageEnd
Ctrl+Backspace	WordDeletePrevious
Ctrl+C	MarkCopyEx
Ctrl+Clear	MoveWordCenter
Ctrl+D	ScrollLineDown
Ctrl+Delete	MarkDeleteEx
Ctrl+E	FileOpenInLine
Ctrl+End	MoveLineEnd
Ctrl+F	SearchDialog
Ctrl+F1	HelpQuickHelp
Ctrl+F5	SearchCaseToggle
Ctrl+F6	SearchRegexpToggle
Ctrl+G	FunctionFindAll
Ctrl+Home	MoveLineHome
Ctrl+I	MarkCopyEx
Ctrl+Insert	MarkCopyEx
Ctrl+K	LineDeleteStart
Ctrl+L	CompilerOutputPrevious
Ctrl+Left	MoveWordPrevious
Ctrl+LineDown	FunctionFindNext
Ctrl+LineUp	FunctionFindPrevious
Ctrl+M	EnterOpen
Ctrl+Minus	FileClose
Ctrl+N	CompilerOutputNext
Ctrl+O	StandardOutputView
Ctrl+P	CompilerOutputView
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+R	MacroRepeat
Ctrl+Return	EnterNext
Ctrl+Right	MoveWordNext
Ctrl+S	ScrollLineUp
Ctrl+T	ScrollLinePageStart
Ctrl+U	Redo
Ctrl+W	FileBackupToggle
Ctrl+X	Redo
Ctrl+Z	Undo
Delete	MarkDeleteEx
End	MoveDocumentEndEx
Escape	MarkHide
F10	ToolsCommand
F5	SearchForward
F6	ReplaceForward
F7	MacroRecordToggle
F8	MacroPlay
F9	MacroLoad
Home	MoveDocumentStartEx
Insert	MarkPasteEx

Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
Minus	MarkCutEx
Multiply	Undo
PageDown	MovePageDown
PageUp	MovePageUp
Plus	MarkCopyEx
Return	Enter
Right	MoveLineRight
Shift+Alt+F1	MacroExecute1
Shift+Alt+F10	MacroExecute10
Shift+Alt+F11	MacroExecute11
Shift+Alt+F12	MacroExecute12
Shift+Alt+F2	MacroExecute2
Shift+Alt+F3	MacroExecute3
Shift+Alt+F4	MacroExecute4
Shift+Alt+F5	MacroExecute5
Shift+Alt+F6	MacroExecute6
Shift+Alt+F7	MacroExecute7
Shift+Alt+F8	MacroExecute8
Shift+Alt+F9	MacroExecute9
Shift+Alt+Left	MoveWordStart
Shift+Alt+Right	MoveWordEnd
Shift+Alt+S	SearchDialog
Shift+Clear	MovePageCenter
Shift+Ctrl+Clear	MoveWordCenter
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F10	ToolsExecute10
Shift+Ctrl+F11	ToolsExecute11
Shift+Ctrl+F12	ToolsExecute12
Shift+Ctrl+F2	ToolsExecute2
Shift+Ctrl+F3	ToolsExecute3
Shift+Ctrl+F4	ToolsExecute4
Shift+Ctrl+F5	ToolsExecute5
Shift+Ctrl+F6	ToolsExecute6
Shift+Ctrl+F7	ToolsExecute7
Shift+Ctrl+F8	ToolsExecute8
Shift+Ctrl+F9	ToolsExecute9
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+LineDown	FunctionFindNext
Shift+Ctrl+LineUp	FunctionFindPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+Delete	MarkCutEx
Shift+End	MoveLineEnd
Shift+F5	SearchNext
Shift+F6	ReplaceNext
Shift+Home	MoveLineHome
Shift+Insert	MarkPasteEx
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Return	EnterOpen

Shift+Right	MoveLineRight
Shift+Tab	TabBack
Tab	Tab

Brief Version 3.10 Keyboard Function Groups

Below is a list of some of the more common Brief commands group by functionality. For a complete list of the Brief keyboard commands listed alphabetically refer to the keyboard mapping section.

General Functions

Alt+F	FileName
Alt+I	InsertModeToggle
Alt+V	Version
Alt+X	FileExit

File Functions

Alt+B	FileListDisplay
Alt+E	FileOpen
Alt+O	FileSaveAs
Alt+P	FilePrint
Alt+R	FileInsertAtCursor
Alt+W	FileSave
Ctrl+-	FileClose
Ctrl+E	FileOpenInLine
Ctrl+Minus	FileClose
Ctrl+W	FileBackupToggle

Navigation Functions

Alt+-	WindowPrevious
Alt+0	BookMarkDrop0
Alt+1	BookMarkDrop1
Alt+2	BookMarkDrop2
Alt+3	BookMarkDrop3
Alt+4	BookMarkDrop4
Alt+5	BookMarkDrop5
Alt+6	BookMarkDrop6
Alt+7	BookMarkDrop7
Alt+8	BookMarkDrop8
Alt+9	BookMarkDrop9
Alt+End	MoveLineRightEdge
Alt+G	LineGoto
Alt+Home	MoveLineLeftEdge
Alt+J	BookMarkGoto
Alt+Left	MoveWordStart
Alt+Minus	WindowPrevious
Alt+N	WindowNext
Alt+Right	MoveWordEnd
Clear	MovePageCenter
Ctrl+Clear	MoveWordCenter
Ctrl+End	MoveLineEnd
Ctrl+Home	MoveLineHome
Ctrl+Left	MoveWordPrevious
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Right	MoveWordNext
End	MoveDocumentEndEx
Home	MoveDocumentStartEx
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp

PageDown	MovePageDown
PageUp	MovePageUp
Right	MoveLineRight
Shift+Alt+Left	MoveWordStart
Shift+Alt+Right	MoveWordEnd
Shift+Clear	MovePageCenter
Shift+Ctrl+Clear	MoveWordCenter
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+End	MoveLineEnd
Shift+Home	MoveLineHome
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Right	MoveLineRight

Undo Functions

Alt+U	Undo
Ctrl+U	Redo
Ctrl+X	Redo
Ctrl+Z	Undo
Multiply	Undo

Mark Functions

Alt+A	MarkColumnToggle
Alt+C	MarkColumnToggle
Alt+H	MarkWordCurrent
Alt+L	MarkLineToggle
Alt+M	MarkBlockToggle
Ctrl+C	MarkCopyEx
Ctrl+Delete	MarkDeleteEx
Ctrl+I	MarkCopyEx
Ctrl+Insert	MarkCopyEx
Delete	MarkDeleteEx
Escape	MarkHide
Insert	MarkPasteEx
Minus	MarkCutEx
Plus	MarkCopyEx
Shift+Delete	MarkCutEx
Shift+Insert	MarkPasteEx

Line Functions

Alt+Backspace	WordDeleteNext
Alt+D	LineDelete
Alt+K	LineDeleteEnd
Backspace	Backspace
Ctrl+Backspace	WordDeletePrevious
Ctrl+K	LineDeleteStart
Ctrl+M	EnterOpen
Ctrl+Return	EnterNext
Return	Enter
Shift+Return	EnterOpen
Shift+Tab	TabBack
Tab	Tab

Search Functions

Alt+F5	SearchReverse
Alt+F6	ReplaceReverse
Alt+S	SearchWordCurrent
Alt+T	ReplaceWordCurrent
Ctrl+F	SearchDialog
Ctrl+F5	SearchCaseToggle
Ctrl+F6	SearchRegexpToggle
F5	SearchForward
F6	ReplaceForward
Shift+Alt+S	SearchDialog
Shift+F5	SearchNext
Shift+F6	ReplaceNext

Scroll Functions

Ctrl+B	ScrollLinePageEnd
Ctrl+D	ScrollLineDown
Ctrl+S	ScrollLineUp
Ctrl+T	ScrollLinePageStart

Build Functions

Alt+F10	CompilerCompile
Ctrl+L	CompilerOutputPrevious
Ctrl+N	CompilerOutputNext
Ctrl+P	CompilerOutputView

Tool Functions

Alt+Z	ToolsShell
Ctrl+O	StandardOutputView
F10	ToolsCommand
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F10	ToolsExecute10
Shift+Ctrl+F11	ToolsExecute11
Shift+Ctrl+F12	ToolsExecute12
Shift+Ctrl+F2	ToolsExecute2
Shift+Ctrl+F3	ToolsExecute3
Shift+Ctrl+F4	ToolsExecute4
Shift+Ctrl+F5	ToolsExecute5
Shift+Ctrl+F6	ToolsExecute6
Shift+Ctrl+F7	ToolsExecute7
Shift+Ctrl+F8	ToolsExecute8
Shift+Ctrl+F9	ToolsExecute9

MacroFunctions

Alt+F7	MacroLoad
Alt+F8	MacroSave
Ctrl+R	MacroRepeat
F7	MacroRecordToggle
F8	MacroPlay
F9	MacroLoad
Shift+Alt+F1	MacroExecute1
Shift+Alt+F10	MacroExecute10
Shift+Alt+F11	MacroExecute11
Shift+Alt+F12	MacroExecute12
Shift+Alt+F2	MacroExecute2
Shift+Alt+F3	MacroExecute3
Shift+Alt+F4	MacroExecute4

Shift+Alt+F5	MacroExecute5
Shift+Alt+F6	MacroExecute6
Shift+Alt+F7	MacroExecute7
Shift+Alt+F8	MacroExecute8
Shift+Alt+F9	MacroExecute9

Special Functions

Ctrl+,	FunctionFindPrevious
Ctrl+.	FunctionFindNext
Ctrl+G	FunctionFindAll
Ctrl+LineDown	FunctionFindNext
Ctrl+LineUp	FunctionFindPrevious
Ctrl+[	BraceMatch
Ctrl+]	BraceMatch
Shift+Ctrl+LineDown	FunctionFindNext
Shift+Ctrl+LineUp	FunctionFindPrevious

Help Functions

Alt+F1	HelpQuickSearch	
Alt+Q	HelpQuickHelp	
Ctrl+F1	HelpQuickHelpF1	HelpIndex

Brief Extended Keyboard

The Brief Extended keyboard mapping can be used to configure Zeus for a Brief keyboard look and feel. This keyboard mapping also adds several extended features designed to make the editor work more effectively in the Windows GUI environment. To find out more information about the keystrokes provided select from one of the items listed below:

Keyboard Mapping

Keyboard Function Groups

Brief Extended Keyboard Mapping

Below is list of keyboard commands provided by the Brief Extended keyboard mapping. Note that some Brief commands that are not supported may be supported in future versions while others commands are just not applicable as they have little or no meaning in a Windows MDI environment. For a list of the more commonly used keystrokes ordered by group refer to the keyboard function groups section.

Alt+-	WindowPrevious
Alt+0	BookMarkDrop0
Alt+1	BookMarkDrop1
Alt+2	BookMarkDrop2
Alt+3	BookMarkDrop3
Alt+4	BookMarkDrop4
Alt+5	BookMarkDrop5
Alt+6	BookMarkDrop6
Alt+7	BookMarkDrop7
Alt+8	BookMarkDrop8
Alt+9	BookMarkDrop9
Alt+A	MarkColumnToggle
Alt+B	FileListDisplay
Alt+Backspace	WordDeleteNext
Alt+C	MarkColumnToggle
Alt+D	LineDelete
Alt+E	FileOpen
Alt+End	MoveLineRightEdge
Alt+F	FileName
Alt+F1	HelpQuickSearch
Alt+F10	CompilerCompile
Alt+F5	SearchReverse
Alt+F6	ReplaceReverse
Alt+F7	MacroLoad
Alt+F8	MacroSave
Alt+G	LineGoto
Alt+H	MarkWordCurrent
Alt+Home	MoveLineLeftEdge
Alt+I	InsertModeToggle
Alt+J	BookMarkGoto
Alt+K	LineDeleteEnd
Alt+L	MarkLineToggle
Alt+Left	MoveWordStart
Alt+M	MarkBlockToggle
Alt+Minus	WindowPrevious
Alt+N	WindowNext
Alt+O	FileSaveAs
Alt+P	FilePrint
Alt+Q	HelpQuickHelp
Alt+R	FileInsertAtCursor
Alt+Right	MoveWordEnd
Alt+S	SearchWordCurrent
Alt+T	ReplaceWordCurrent
Alt+U	Undo
Alt+V	Version
Alt+W	FileSave
Alt+X	FileExit

Alt+Z	ToolsShell
Backspace	Backspace
Clear	MovePageCenter
Ctrl+,	FunctionFindPrevious
Ctrl+-	FileClose
Ctrl+.	FunctionFindNext
Ctrl+[	BraceMatch
Ctrl+]	BraceMatch
Ctrl+B	ScrollLinePageEnd
Ctrl+Backspace	WordDeletePrevious
Ctrl+C	MarkCopyEx
Ctrl+Clear	MoveWordCenter
Ctrl+D	ScrollLineDown
Ctrl+Delete	MarkDeleteEx
Ctrl+E	FileOpenInLine
Ctrl+End	MoveLineEnd
Ctrl+F	SearchDialog
Ctrl+F1	HelpQuickHelp
Ctrl+F5	SearchCaseToggle
Ctrl+F6	SearchRegexpToggle
Ctrl+G	FunctionFindAll
Ctrl+Home	MoveLineHome
Ctrl+I	MarkCopyEx
Ctrl+Insert	MarkCopyEx
Ctrl+K	LineDeleteStart
Ctrl+L	CompilerOutputPrevious
Ctrl+Left	MoveWordPrevious
Ctrl+LineDown	FunctionFindNext
Ctrl+LineUp	FunctionFindPrevious
Ctrl+M	EnterOpen
Ctrl+Minus	FileClose
Ctrl+N	CompilerOutputNext
Ctrl+O	StandardOutputView
Ctrl+P	CompilerOutputView
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+R	MacroRepeat
Ctrl+Return	EnterNext
Ctrl+Right	MoveWordNext
Ctrl+S	ScrollLineUp
Ctrl+T	ScrollLinePageStart
Ctrl+U	Redo
Ctrl+W	FileBackupToggle
Ctrl+X	Redo
Ctrl+Z	Undo
Delete	MarkDeleteEx
End	MoveDocumentEndEx
Escape	MarkHide
F1	HelpIndex
F10	ToolsCommand
F5	SearchForward
F6	ReplaceForward
F7	MacroRecordToggle
F8	MacroPlay
F9	MacroLoad

Home	MoveDocumentStartEx
Insert	MarkPasteEx
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
Minus	MarkCutEx
Multiply	Undo
PageDown	MovePageDown
PageUp	MovePageUp
Plus	MarkCopyEx
Return	Enter
Right	MoveLineRight
Shift+Alt+F1	MacroExecute1
Shift+Alt+F10	MacroExecute10
Shift+Alt+F11	MacroExecute11
Shift+Alt+F12	MacroExecute12
Shift+Alt+F2	MacroExecute2
Shift+Alt+F3	MacroExecute3
Shift+Alt+F4	MacroExecute4
Shift+Alt+F5	MacroExecute5
Shift+Alt+F6	MacroExecute6
Shift+Alt+F7	MacroExecute7
Shift+Alt+F8	MacroExecute8
Shift+Alt+F9	MacroExecute9
Shift+Alt+Left	MoveWordStart
Shift+Alt+Right	MoveWordEnd
Shift+Alt+S	SearchDialog
Shift+Clear	MovePageCenter
Shift+Ctrl+Clear	MoveWordCenter
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F10	ToolsExecute10
Shift+Ctrl+F11	ToolsExecute11
Shift+Ctrl+F12	ToolsExecute12
Shift+Ctrl+F2	ToolsExecute2
Shift+Ctrl+F3	ToolsExecute3
Shift+Ctrl+F4	ToolsExecute4
Shift+Ctrl+F5	ToolsExecute5
Shift+Ctrl+F6	ToolsExecute6
Shift+Ctrl+F7	ToolsExecute7
Shift+Ctrl+F8	ToolsExecute8
Shift+Ctrl+F9	ToolsExecute9
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+LineDown	FunctionFindNext
Shift+Ctrl+LineUp	FunctionFindPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+Delete	MarkCutEx
Shift+End	MoveLineEnd
Shift+F5	SearchNext
Shift+F6	ReplaceNext
Shift+Home	MoveLineHome
Shift+Insert	MarkPasteEx
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown

Shift+PageUp	MovePageUp
Shift+Return	EnterOpen
Shift+Right	MoveLineRight
Shift+Tab	TabBack
Tab	Tab

Brief Extended Keyboard Function Groups

Below is a list of some of the more common Brief commands group by functionality. For a complete list of the Brief keyboard commands ordered alphabetically refer to the keyboard mapping section.

General Functions

Alt+F	FileName
Alt+I	InsertModeToggle
Alt+V	Version
Alt+X	FileExit

File Functions

Alt+B	FileListDisplay
Alt+E	FileOpen
Alt+O	FileSaveAs
Alt+P	FilePrint
Alt+R	FileInsertAtCursor
Alt+W	FileSave
Ctrl+-	FileClose
Ctrl+E	FileOpenInLine
Ctrl+Minus	FileClose
Ctrl+W	FileBackupToggle

Navigation Functions

Alt+-	WindowPrevious
Alt+0	BookMarkDrop0
Alt+1	BookMarkDrop1
Alt+2	BookMarkDrop2
Alt+3	BookMarkDrop3
Alt+4	BookMarkDrop4
Alt+5	BookMarkDrop5
Alt+6	BookMarkDrop6
Alt+7	BookMarkDrop7
Alt+8	BookMarkDrop8
Alt+9	BookMarkDrop9
Alt+End	MoveLineRightEdge
Alt+G	LineGoto
Alt+Home	MoveLineLeftEdge
Alt+J	BookMarkGoto
Alt+Left	MoveWordStart
Alt+Minus	WindowPrevious
Alt+N	WindowNext
Alt+Right	MoveWordEnd
Clear	MovePageCenter
Ctrl+Clear	MoveWordCenter
Ctrl+End	MoveLineEnd
Ctrl+Home	MoveLineHome
Ctrl+Left	MoveWordPrevious
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Right	MoveWordNext
End	MoveDocumentEndEx
Home	MoveDocumentStartEx
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp

PageDown	MovePageDown
PageUp	MovePageUp
Right	MoveLineRight
Shift+Alt+Left	MoveWordStart
Shift+Alt+Right	MoveWordEnd
Shift+Clear	MovePageCenter
Shift+Ctrl+Clear	MoveWordCenter
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+End	MoveLineEnd
Shift+Home	MoveLineHome
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Right	MoveLineRight

Undo Functions

Alt+U	Undo
Ctrl+U	Redo
Ctrl+X	Redo
Ctrl+Z	Undo
Multiply	Undo

Mark Functions

Alt+A	MarkColumnToggle
Alt+C	MarkColumnToggle
Alt+H	MarkWordCurrent
Alt+L	MarkLineToggle
Alt+M	MarkBlockToggle
Ctrl+C	MarkCopyEx
Ctrl+Delete	MarkDeleteEx
Ctrl+I	MarkCopyEx
Ctrl+Insert	MarkCopyEx
Delete	MarkDeleteEx
Escape	MarkHide
Insert	MarkPasteEx
Minus	MarkCutEx
Plus	MarkCopyEx
Shift+Delete	MarkCutEx
Shift+Insert	MarkPasteEx

Line Functions

Alt+Backspace	WordDeleteNext
Alt+D	LineDelete
Alt+K	LineDeleteEnd
Backspace	Backspace
Ctrl+Backspace	WordDeletePrevious
Ctrl+K	LineDeleteStart
Ctrl+M	EnterOpen
Ctrl+Return	EnterNext
Return	Enter
Shift+Return	EnterOpen
Shift+Tab	TabBack
Tab	Tab

Search Functions

Alt+F5	SearchReverse
Alt+F6	ReplaceReverse
Alt+S	SearchWordCurrent
Alt+T	ReplaceWordCurrent
Ctrl+F	SearchDialog
Ctrl+F5	SearchCaseToggle
Ctrl+F6	SearchRegexpToggle
F5	SearchForward
F6	ReplaceForward
Shift+Alt+S	SearchDialog
Shift+F5	SearchNext
Shift+F6	ReplaceNext

Scroll Functions

Ctrl+B	ScrollLinePageEnd
Ctrl+D	ScrollLineDown
Ctrl+S	ScrollLineUp
Ctrl+T	ScrollLinePageStart

Build Functions

Alt+F10	CompilerCompile
Ctrl+L	CompilerOutputPrevious
Ctrl+N	CompilerOutputNext
Ctrl+P	CompilerOutputView

Tool Functions

Alt+Z	ToolsShell
Ctrl+O	StandardOutputView
F10	ToolsCommand
Shift+Ctrl+F1	ToolsExecute1
Shift+Ctrl+F10	ToolsExecute10
Shift+Ctrl+F11	ToolsExecute11
Shift+Ctrl+F12	ToolsExecute12
Shift+Ctrl+F2	ToolsExecute2
Shift+Ctrl+F3	ToolsExecute3
Shift+Ctrl+F4	ToolsExecute4
Shift+Ctrl+F5	ToolsExecute5
Shift+Ctrl+F6	ToolsExecute6
Shift+Ctrl+F7	ToolsExecute7
Shift+Ctrl+F8	ToolsExecute8
Shift+Ctrl+F9	ToolsExecute9

Macro Functions

Alt+F7	MacroLoad
Alt+F8	MacroSave
Ctrl+R	MacroRepeat
F7	MacroRecordToggle
F8	MacroPlay
F9	MacroLoad
Shift+Alt+F1	MacroExecute1
Shift+Alt+F10	MacroExecute10
Shift+Alt+F11	MacroExecute11
Shift+Alt+F12	MacroExecute12
Shift+Alt+F2	MacroExecute2
Shift+Alt+F3	MacroExecute3
Shift+Alt+F4	MacroExecute4

Shift+Alt+F5	MacroExecute5
Shift+Alt+F6	MacroExecute6
Shift+Alt+F7	MacroExecute7
Shift+Alt+F8	MacroExecute8
Shift+Alt+F9	MacroExecute9

Special Functions

Ctrl+,	FunctionFindPrevious
Ctrl+.	FunctionFindNext
Ctrl+G	FunctionFindAll
Ctrl+LineDown	FunctionFindNext
Ctrl+LineUp	FunctionFindPrevious
Ctrl+[	BraceMatch
Ctrl+]	BraceMatch
Shift+Ctrl+LineDown	FunctionFindNext
Shift+Ctrl+LineUp	FunctionFindPrevious

Help Functions

Alt+F1	HelpQuickSearch
Alt+Q	HelpQuickHelp
Ctrl+F1	HelpQuickHelp
F1	HelpIndex

Epsilon Keyboard

The Epsilon keyboard mapping can be used to configure Zeus for a Epsilon keyboard look and feel. To find out more information about the keystrokes provided select from one of the items listed below:

Epsilon Keyboard Mapping

Below is list of keyboard commands provided by the Epsilon keyboard mapping. Note that some Epsilon commands that are not supported may be supported in future versions while others commands are just not applicable as they have little or no meaning in a Windows MDI environment. For a list of the more commonly used keystrokes ordered by group refer to the keyboard function groups section.

Alt+/	HelpIndex
Alt+[	MoveLineRight
Alt+]	MoveLineLeft
Alt+0	BraceMatch
Alt+B	MoveWordPrevious
Alt+Backspace	WordDeletePrevious
Alt+Ctrl+S	SearchNext
Alt+E	FileOpen
Alt+End	WindowNext
Alt+F	MoveWordNext
Alt+F1	HelpQuickSearch
Alt+Home	WindowPrevious
Alt+Left	MoveLineHome
Alt+M	MarkBlockToggle
Alt+N	WindowNext
Alt+O	FileSaveAs
Alt+P	WindowPrevious
Alt+Q	HelpQuickHelp
Alt+Right	MoveLineEnd
Alt+S	MovePageCenter
Alt+T	ReplaceWordCurrent
Alt+V	MovePageUp
Alt+Z	MoveLineDown
Backspace	Backspace
Clear	MovePageCenter
Ctrl+A	MoveLineHome
Ctrl+B	MoveLineLeft
Ctrl+Backspace	WordDeletePrevious
Ctrl+C	MarkCopyEx
Ctrl+D	CharDelete
Ctrl+Delete	MarkDeleteEx
Ctrl+E	MoveLineEnd
Ctrl+End	MoveDocumentEnd
Ctrl+F	MoveLineRight
Ctrl+F1	HelpQuickHelp
Ctrl+F10	Redo
Ctrl+F4	MacroPlay
Ctrl+F7	FileSaveAs
Ctrl+F9	Undo
Ctrl+H	Backspace
Ctrl+Home	MoveDocumentStart
Ctrl+Insert	MarkCopyEx
Ctrl+K	LineDelete
Ctrl+L	MovePageCenter
Ctrl+Left	MoveWordPrevious
Ctrl+M	Enter
Ctrl+N	MoveLineDown

Ctrl+O	EnterOpen
Ctrl+P	MoveLineUp
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Return	EnterNext
Ctrl+Right	MoveWordNext
Ctrl+S	SearchNext
Ctrl+U	Redo
Ctrl+V	MovePageDown
Ctrl+W	MarkDelete
Ctrl+X	Redo
Ctrl+X+B	FileListDisplay
Ctrl+X+Ctrl+C	FileExit
Ctrl+X+Ctrl+F	FileOpen
Ctrl+X+Ctrl+I	Tab
Ctrl+X+Ctrl+R	Redo
Ctrl+X+Ctrl+Tab	Tab
Ctrl+X+Ctrl+U	Undo
Ctrl+X+Ctrl+W	FileSave
Ctrl+X+E	MacroPlay
Ctrl+X+G	LineGoto
Ctrl+X+K	FileClose
Ctrl+X+M	CompilerCompile
Ctrl+X+N	WindowNext
Ctrl+X+P	WindowPrevious
Ctrl+X+R	Redo
Ctrl+X+S	FileSaveAll
Ctrl+X+Tab	Tab
Ctrl+X+U	Undo
Ctrl+Z	MoveLineDown
Delete	MarkDeleteEx
Escape	MarkHide
F1	HelpIndex
F10	Redo
F9	Undo
Insert	InsertModeToggle
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
PageDown	MovePageDown
PageUp	MovePageUp
Return	Enter
Right	MoveLineRight
Shift+Ctrl+-	HelpIndex
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+Delete	MarkCutEx
Shift+End	MoveLineEnd
Shift+F1	ToolsExecute1
Shift+F10	ToolsExecute9
Shift+F11	ToolsExecute10
Shift+F12	ToolsExecute11
Shift+F13	ToolsExecute12
Shift+F14	ToolsExecute13
Shift+F15	ToolsExecute14

Shift+F16	ToolsExecute15
Shift+F2	ToolsExecute3
Shift+F3	ToolsExecute4
Shift+F4	ToolsExecute6
Shift+F5	SearchNext
Shift+F8	ToolsExecute7
Shift+F9	ToolsExecute8
Shift+Home	MoveLineHome
Shift+Insert	MarkPasteEx
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Return	EnterOpen
Shift+Right	MoveLineRight
Shift+Tab	TabBack
Tab	Tab

Epsilon Keyboard Function Groups

Below is a list of some of the more common Epsilon commands group by functionality. For a complete list of the Epsilon keyboard commands ordered alphabetically refer to the keyboard mapping section.

General Functions

Ctrl+X+Ctrl+C	FileExit
Insert	InsertModeToggle

File Functions

Alt+E	FileOpen
Alt+O	FileSaveAs
Ctrl+F7	FileSaveAs
Ctrl+X+B	FileListDisplay
Ctrl+X+Ctrl+F	FileOpen
Ctrl+X+Ctrl+W	FileSave
Ctrl+X+K	FileClose
Ctrl+X+S	FileSaveAll

Navigation Functions

Alt+B	MoveWordPrevious
Alt+End	WindowNext
Alt+F	MoveWordNext
Alt+Home	WindowPrevious
Alt+Left	MoveLineHome
Alt+N	WindowNext
Alt+P	WindowPrevious
Alt+Right	MoveLineEnd
Alt+S	MovePageCenter
Alt+V	MovePageUp
Alt+Z	MoveLineDown
Alt+[	MoveLineRight
Alt+]	MoveLineLeft
Clear	MovePageCenter
Ctrl+A	MoveLineHome
Ctrl+B	MoveLineLeft
Ctrl+E	MoveLineEnd
Ctrl+End	MoveDocumentEnd
Ctrl+F	MoveLineRight
Ctrl+Home	MoveDocumentStart
Ctrl+L	MovePageCenter
Ctrl+Left	MoveWordPrevious
Ctrl+N	MoveLineDown
Ctrl+P	MoveLineUp
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Right	MoveWordNext
Ctrl+V	MovePageDown
Ctrl+X+G	LineGoto
Ctrl+X+N	WindowNext
Ctrl+X+P	WindowPrevious
Ctrl+Z	MoveLineDown
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
PageDown	MovePageDown

PageUp	MovePageUp
Right	MoveLineRight
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+End	MoveLineEnd
Shift+Home	MoveLineHome
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Right	MoveLineRight

Undo/Redo Functions

Ctrl+F10	Redo
Ctrl+F9	Undo
Ctrl+U	Redo
Ctrl+X	Redo
Ctrl+X+Ctrl+R	Redo
Ctrl+X+Ctrl+U	Undo
Ctrl+X+R	Redo
Ctrl+X+U	Undo
F10	Redo
F9	Undo

Mark Functions

Alt+M	MarkBlockToggle
Ctrl+C	MarkCopyEx
Ctrl+Delete	MarkDeleteEx
Ctrl+Insert	MarkCopyEx
Ctrl+W	MarkDelete
Delete	MarkDeleteEx
Escape	MarkHide
Shift+Delete	MarkCutEx
Shift+Insert	MarkPasteEx

Line Functions

Alt+Backspace	WordDeletePrevious
Backspace	Backspace
Ctrl+Backspace	WordDeletePrevious
Ctrl+D	CharDelete
Ctrl+H	Backspace
Ctrl+K	LineDelete
Ctrl+M	Enter
Ctrl+O	EnterOpen
Ctrl+Return	EnterNext
Ctrl+X+Ctrl+I	Tab
Ctrl+X+Ctrl+Tab	Tab
Ctrl+X+Tab	Tab
Return	Enter
Shift+Return	EnterOpen
Shift+Tab	TabBack
Tab	Tab

Search Functions

Alt+Ctrl+S	SearchNext
Alt+T	ReplaceWordCurrent

Ctrl+S	SearchNext
Shift+F5	SearchNext

Build Functions

Ctrl+X+M	CompilerCompile

Tool Functions

Shift+F1	ToolsExecute1
Shift+F10	ToolsExecute9
Shift+F11	ToolsExecute10
Shift+F12	ToolsExecute11
Shift+F13	ToolsExecute12
Shift+F14	ToolsExecute13
Shift+F15	ToolsExecute14
Shift+F16	ToolsExecute15
Shift+F2	ToolsExecute3
Shift+F3	ToolsExecute4
Shift+F4	ToolsExecute6
Shift+F8	ToolsExecute7
Shift+F9	ToolsExecute8

Macro Functions

Ctrl+F4	MacroPlay
Ctrl+X+E	MacroPlay

SpecialFunctions

Alt+0	BraceMatch

Help Functions

Alt+/	HelpIndex
Alt+F1	HelpQuickSearch
Alt+Q	HelpQuickHelp
Ctrl+F1	HelpQuickHelp
F1	HelpIndex
Shift+Ctrl+-	HelpIndex

WordStar Keyboard

The WordStar keyboard mapping can be used to configure Zeus for a WordStar keyboard look and feel. To find out more information about the keystrokes provided select from one of the items listed below:

WordStar Keyboard Mapping

Below is list of keyboard commands provided by the WordStar keyboard mapping. Note that some WordStar commands that are not supported may be supported in future versions while others commands are just not applicable as they have little or no meaning in a Windows MDI environment. For a list of the more commonly used keystrokes ordered by group refer to the keyboard function groups section.

Alt+Backspace	Undo
Alt+F3	FileClose
Alt+F9	CompilerCompile
Alt+Q	HelpQuickHelp
Backspace	Backspace
Clear	MovePageCenter
Ctrl+A	MoveWordPrevious
Ctrl+Backspace	WordDeletePrevious
Ctrl+C	MovePageDown
Ctrl+Delete	MarkDeleteEx
Ctrl+E	MoveLineUp
Ctrl+End	MoveLineEnd
Ctrl+F	MoveWordNext
Ctrl+F1	HelpQuickHelp
Ctrl+F1	HelpQuickSearch
Ctrl+G	CharDelete
Ctrl+H	Backspace
Ctrl+I	Tab
Ctrl+Insert	MarkCopyEx
Ctrl+K+0	BookMarkDrop0
Ctrl+K+1	BookMarkDrop1
Ctrl+K+2	BookMarkDrop2
Ctrl+K+3	BookMarkDrop3
Ctrl+K+4	BookMarkDrop4
Ctrl+K+5	BookMarkDrop5
Ctrl+K+6	BookMarkDrop6
Ctrl+K+7	BookMarkDrop7
Ctrl+K+8	BookMarkDrop8
Ctrl+K+9	BookMarkDrop9
Ctrl+K+Ctrl+0	BookMarkDrop0
Ctrl+K+Ctrl+1	BookMarkDrop1
Ctrl+K+Ctrl+C	MarkCopy
Ctrl+K+Ctrl+H	MarkHide
Ctrl+K+Ctrl+I	Tab
Ctrl+K+Ctrl+L	MarkLineToggle
Ctrl+K+Ctrl+S	FileSave
Ctrl+K+Ctrl+T	MarkWordCurrent
Ctrl+K+Ctrl+U	TabBack
Ctrl+K+Ctrl+Y	MarkDelete
Ctrl+K+Shift+0	BookMarkDrop0
Ctrl+K+Shift+2	BookMarkDrop2
Ctrl+K+Shift+3	BookMarkDrop3
Ctrl+K+Shift+4	BookMarkDrop4
Ctrl+K+Shift+5	BookMarkDrop5
Ctrl+K+Shift+6	BookMarkDrop6
Ctrl+K+Shift+7	BookMarkDrop7
Ctrl+K+Shift+8	BookMarkDrop8

Ctrl+K+Shift+9	BookMarkDrop9
Ctrl+L	SearchNext
Ctrl+Left	MoveWordPrevious
Ctrl+N	EnterOpen
Ctrl+O+Ctrl+A	ReplaceWordCurrent
Ctrl+O+Ctrl+C	MarkColumnToggle
Ctrl+O+Ctrl+F	SearchWordCurrent
Ctrl+O+Ctrl+G	LineGoto
Ctrl+O+Ctrl+I	MarkBlockToggle
Ctrl+O+Ctrl+L	MarkLineToggle
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Q+0	BookMarkGoto0
Ctrl+Q+1	BookMarkGoto1
Ctrl+Q+2	BookMarkGoto2
Ctrl+Q+3	BookMarkGoto3
Ctrl+Q+4	BookMarkGoto4
Ctrl+Q+5	BookMarkGoto5
Ctrl+Q+6	BookMarkGoto6
Ctrl+Q+7	BookMarkGoto7
Ctrl+Q+8	BookMarkGoto8
Ctrl+Q+9	BookMarkGoto9
Ctrl+Q+Ctrl+0	BookMarkGoto0
Ctrl+Q+Ctrl+1	BookMarkGoto1
Ctrl+Q+Ctrl+2	BookMarkGoto2
Ctrl+Q+Ctrl+3	BookMarkGoto3
Ctrl+Q+Ctrl+4	BookMarkGoto4
Ctrl+Q+Ctrl+5	BookMarkGoto5
Ctrl+Q+Ctrl+6	BookMarkGoto6
Ctrl+Q+Ctrl+7	BookMarkGoto7
Ctrl+Q+Ctrl+8	BookMarkGoto8
Ctrl+Q+Ctrl+9	BookMarkGoto9
Ctrl+Q+Ctrl+A	ReplaceWordCurrent
Ctrl+Q+Ctrl+C	MoveDocumentEnd
Ctrl+Q+Ctrl+D	MoveLineEnd
Ctrl+Q+Ctrl+F	SearchWordCurrent
Ctrl+Q+Ctrl+R	MoveDocumentStart
Ctrl+Q+Ctrl+S	MoveLineHome
Ctrl+Q+Ctrl+T	ScrollLinePageStart
Ctrl+Q+Ctrl+U	ScrollLinePageEnd
Ctrl+Q+Ctrl+Y	LineDeleteEnd
Ctrl+R	MovePageUp
Ctrl+Right	MoveWordNext
Ctrl+V	InsertModeToggle
Ctrl+W	MoveLineUp
Ctrl+X	MoveLineDown
Ctrl+Y	LineDelete
Ctrl+Z	MoveLineDown
Delete	CharDelete
End	MoveLineEnd
Escape	MarkHide
F1	HelpIndex
F2	FileSave
F3	FileOpen
F6	WindowNext

Home	MoveLineHome
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
PageDown	MovePageDown
PageUp	MovePageUp
Right	MoveLineRight
Shift+Alt+Backspace	Redo
Shift+Backspace	Backspace
Shift+Ctrl+A	MoveWordPrevious
Shift+Ctrl+F	MoveWordNext
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+P	MacroPlay
Shift+Ctrl+Right	MoveWordNext
Shift+Delete	MarkCutEx
Shift+End	MoveLineEnd
Shift+Home	MoveLineHome
Shift+Insert	MarkPasteEx
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Right	MoveLineRight
Shift+Tab	TabBack
Tab	Tab

WordStar Keyboard Function Groups

Below is a list of some of the more common WordStar commands group by functionality. For a complete list of the WordStar keyboard commands ordered alphabetically refer to the keyboard mapping section.

General Functions

Ctrl+V	InsertModeToggle

File Functions

Alt+F3	FileClose
Ctrl+K+Ctrl+S	FileSave
F2	FileSave
F3	FileOpen

Naviagtion Functions

Clear	MovePageCenter
Ctrl+A	MoveWordPrevious
Ctrl+C	MovePageDown
Ctrl+E	MoveLineUp
Ctrl+End	MoveLineEnd
Ctrl+F	MoveWordNext
Ctrl+K+0	BookMarkDrop0
Ctrl+K+1	BookMarkDrop1
Ctrl+K+2	BookMarkDrop2
Ctrl+K+3	BookMarkDrop3
Ctrl+K+4	BookMarkDrop4
Ctrl+K+5	BookMarkDrop5
Ctrl+K+6	BookMarkDrop6
Ctrl+K+7	BookMarkDrop7
Ctrl+K+8	BookMarkDrop8
Ctrl+K+9	BookMarkDrop9
Ctrl+K+Ctrl+0	BookMarkDrop0
Ctrl+K+Ctrl+1	BookMarkDrop1
Ctrl+K+Shift+0	BookMarkDrop0
Ctrl+K+Shift+2	BookMarkDrop2
Ctrl+K+Shift+3	BookMarkDrop3
Ctrl+K+Shift+4	BookMarkDrop4
Ctrl+K+Shift+5	BookMarkDrop5
Ctrl+K+Shift+6	BookMarkDrop6
Ctrl+K+Shift+7	BookMarkDrop7
Ctrl+K+Shift+8	BookMarkDrop8
Ctrl+K+Shift+9	BookMarkDrop9
Ctrl+Left	MoveWordPrevious
Ctrl+O+Ctrl+G	LineGoto
Ctrl+PageDown	MoveDocumentEnd
Ctrl+PageUp	MoveDocumentStart
Ctrl+Q+0	BookMarkGoto0
Ctrl+Q+1	BookMarkGoto1
Ctrl+Q+2	BookMarkGoto2
Ctrl+Q+3	BookMarkGoto3
Ctrl+Q+4	BookMarkGoto4
Ctrl+Q+5	BookMarkGoto5
Ctrl+Q+6	BookMarkGoto6
Ctrl+Q+7	BookMarkGoto7
Ctrl+Q+8	BookMarkGoto8

Ctrl+Q+9	BookMarkGoto9
Ctrl+Q+Ctrl+0	BookMarkGoto0
Ctrl+Q+Ctrl+1	BookMarkGoto1
Ctrl+Q+Ctrl+2	BookMarkGoto2
Ctrl+Q+Ctrl+3	BookMarkGoto3
Ctrl+Q+Ctrl+4	BookMarkGoto4
Ctrl+Q+Ctrl+5	BookMarkGoto5
Ctrl+Q+Ctrl+6	BookMarkGoto6
Ctrl+Q+Ctrl+7	BookMarkGoto7
Ctrl+Q+Ctrl+8	BookMarkGoto8
Ctrl+Q+Ctrl+9	BookMarkGoto9
Ctrl+Q+Ctrl+C	MoveDocumentEnd
Ctrl+Q+Ctrl+D	MoveLineEnd
Ctrl+Q+Ctrl+R	MoveDocumentStart
Ctrl+Q+Ctrl+S	MoveLineHome
Ctrl+R	MovePageUp
Ctrl+Right	MoveWordNext
Ctrl+W	MoveLineUp
Ctrl+X	MoveLineDown
Ctrl+Z	MoveLineDown
End	MoveLineEnd
F6	WindowNext
Home	MoveLineHome
Left	MoveLineLeft
LineDown	MoveLineDown
LineUp	MoveLineUp
PageDown	MovePageDown
PageUp	MovePageUp
Right	MoveLineRight
Shift+Ctrl+A	MoveWordPrevious
Shift+Ctrl+F	MoveWordNext
Shift+Ctrl+Left	MoveWordPrevious
Shift+Ctrl+Right	MoveWordNext
Shift+End	MoveLineEnd
Shift+Home	MoveLineHome
Shift+Left	MoveLineLeft
Shift+LineDown	MoveLineDown
Shift+LineUp	MoveLineUp
Shift+PageDown	MovePageDown
Shift+PageUp	MovePageUp
Shift+Right	MoveLineRight

Undo Functions

Alt+Backspace	Undo
Shift+Alt+Backspace	Redo

Mark Functions

Ctrl+Delete	MarkDeleteEx
Ctrl+Insert	MarkCopyEx
Ctrl+K+Ctrl+C	MarkCopy
Ctrl+K+Ctrl+H	MarkHide
Ctrl+K+Ctrl+L	MarkLineToggle
Ctrl+K+Ctrl+T	MarkWordCurrent
Ctrl+K+Ctrl+Y	MarkDelete
Ctrl+O+Ctrl+C	MarkColumnToggle
Ctrl+O+Ctrl+I	MarkBlockToggle

Ctrl+O+Ctrl+L	MarkLineToggle
Escape	MarkHide
Shift+Delete	MarkCutEx
Shift+Insert	MarkPasteEx

Line Functions

Backspace	Backspace
Ctrl+Backspace	WordDeletePrevious
Ctrl+G	CharDelete
Ctrl+H	Backspace
Ctrl+I	Tab
Ctrl+K+Ctrl+I	Tab
Ctrl+K+Ctrl+U	TabBack
Ctrl+N	EnterOpen
Ctrl+Q+Ctrl+Y	LineDeleteEnd
Ctrl+Y	LineDelete
Delete	CharDelete
Shift+Backspace	Backspace
Shift+Tab	TabBack
Tab	Tab

Search Functions

Ctrl+L	SearchNext
Ctrl+O+Ctrl+A	ReplaceWordCurrent
Ctrl+O+Ctrl+F	SearchWordCurrent
Ctrl+Q+Ctrl+A	ReplaceWordCurrent
Ctrl+Q+Ctrl+F	SearchWordCurrent
Ctrl+Q+Ctrl+T	ScrollLinePageStart
Ctrl+Q+Ctrl+U	ScrollLinePageEnd

Build Functions

Alt+F9	CompilerCompile

Macro Functions

Shift+Ctrl+P	MacroPlay

Help Functions

Alt+Q	HelpQuickHelp
Ctrl+F1	HelpQuickHelp
Ctrl+F1	HelpQuickSearch
F1	HelpIndex

Keyboard Options Dialog

This dialog is used configure the way Zeus handles the user keyboard input. To use this dialog you will need to either create a new keyboard mapping or edit one of the existing keyboard mappings. Alternatively you can apply one of the pre-defined keyboard definitions as the currently active keyboard mapping. For more information on each of the different sections of this dialog click on one of the items listed below:

- New
- Edit
- Delete
- Import
- Export
- Apply
- Cancel
- Help

Also refer to the New Keyboard Dialog and Edit Keyboard Dialog sections for more information about configuring the Zeus keyboard handler.

New

Create a new keyboard mapping with the name as specified in the new keymap edit field.

Edit

Edit the keyboard mapping that is currently selected in the Defined Keymaps list.

Delete

Delete the keyboard mapping that is currently selected in the Defined Keymaps list.

Import

Import a keyboard mapping from a keyboard definition file. This feature is offered as a method of restoring a keyboard definition that had been saved earlier using the export feature.

Export

Export the keyboard map that is currently selected in the Defined Keymaps list to keyboard definition file. This feature is offered as a method of backing up the keyboard mapping, as the file produced guaranteed to be upwardly compatible for all versions of the Zeus editor.

Apply

Make the keyboard mapping selected in the Defined Keymaps list the currently active keyboard mapping.

Functions Available

This list all the functions that are available for binding to the specific key.

New Keymap Dialog

This dialog is used create a new keyboard mapping. To create the keyboard mapping set the focus to the Prefix Key edit field using the mouse (or by tabbing to the entry field) and type in the key combination required. The dialog will indicate if the key is already bound (or is not bound) by showing a highlighted function in the list of functions available and by displaying a message in the status line. At this point you may the either add, remove or modify the binding for the key specified. To change the key binding just select a new function from the list of functions available.

Note that the TAB and Shift-TAB keys are reserved for use by Windows so to bind these keys you will need to manually select the VK_TAB from the keycode list. Also remember that Zeus binds keys as a single unit. This means that the Alt + W key binding is not the same as the Alt key followed by the W key.

For more information on each of the different sections of this dialog click on one of the items listed below:

- Allow CUA Marking
- Prefix Key
- Key Code
- Extended Code
- Functions Available
- Update
- Remove
- Cancel
- Help

Edit Keymap Dialog

This dialog is used edit an existing keyboard mapping. To edit the keyboard mapping set the focus to the Prefix Key edit field using the mouse (or by tabbing to the entry field) and type in the key combination required. The dialog will indicate if the key is already bound (or is not bound) by showing a highlighted function in the list of functions available and by displaying a message in the status line. At this point you may the either add, remove or modify the binding for the key specified. To change the key binding just select a new function from the list of functions available.

Note that the TAB and Shift-TAB keys are reserved for use by Windows so to bind these keys you will need to manually select the VK_TAB from the keycode list. Also remember that Zeus binds keys as a single unit. This means that the Alt + W key binding is not the same as the Alt key followed by the W key.

For more information on each of the different sections of this dialog click on one of the items listed below:

- Allow CUA Marking
- Key Press Removes Mark
- Cut Copy Removes Mark
- Prefix Key
- Key Code
- Extended Code
- Functions Available
- Update
- Remove
- Cancel
- Help

Allow CUA Marking

Enable the CUA marking for this keyboard mapping. This only enables or disables the CUA marking. To get the CUA functions to work properly you will also need to bind the functions for all the CUA keystrokes. For example for the case of the CUA line end marking (ie Shift + End) the Shift + End keyboard combination will also need to be bound to the MoveLineEnd function. Note that when CUA marking is enabled any time a key is pressed or any if any of the cursor navigation keys are used then this will result in the marked area being removed.

Key Press Removes Mark

When enabled this option will force Zeus to replace any marked area with the key just pressed and in the process remove the marked area. Note that if you disable this feature the a key press will never replace a marked are but the marked are may or may not be removed depending on whether the CUA marking option is enabled.

Cut Copy Removes Mark

When enabled this option will remove the marked area any time a clipboard cut or copy operation is performed.

Prefix Key

Set the prefix key to be used. This allows prefix key combinations to be defined. For example to define a binding for Ctrl-X Ctrl-A first define the prefix to be Ctrl-X and then binding a function to the Ctrl-A keystroke.

Key Code

This is the virtual keycode of the key pressed. Note that for tab and shift tab keys you will need to select the VK_TAB virtual key code manually from the list.

Extended Code

This is the extended key information of the key pressed. This information defines if the Alt, Shift or Ctrl keys are required to fully define the keystroke. Note that in the case of CUA keys the Shift key should always be defined otherwise the CUA marking will not update the screen. Also remember that Zeus binds keys as a single unit. This means that the Alt + W key binding is not the same as the Alt key followed by the W key.

Update

Update the changes made to the keyboard mapping.

Remove

Remove the binding for the function selected in the Functions Available list.

Standard 101 Keyboard

The Zeus editor performs all of its keyboard handling based on the scan codes generated by the keyboard. Windows helps in this regard as most of the keys on a standard 101 keyboard are mapped to virtual Windows key codes and these are in fact common across all languages that are supported by Windows. The exception to this rule is a hand full of special keys that are language specific. These keys change meaning depending on the currently loaded code page and country information (as configured by the control panel international settings).

As such these keys can not be easily identified using an ASCII character pneumonic. For this reason Zeus identifies these characters using their scan codes and it does so by adding to the Windows list of virtual keys. These additional virtual key codes and the list of the keys that Zeus defines for this purpose are listed below:

Pneumonic	Scan Code	ASCII	HEX CODE
VK_0xC0	0x0C0	'~'	C0
VK_0xBD	0x0BD	'-'	BD
VK_0xBB	0x0BB	'='	BB
VK_0xDC	0x0DC	'\'	DC
VK_0xDB	0x0DB	'['	DB
VK_0xDD	0x0DD	']'	DD
VK_0xBA	0x0BA	';'	BA
VK_0xDE	0x0DE	'''	DE
VK_0xBC	0x0BC	','	BC
VK_0xBE	0x0BE	'.'	BE
VK_0xBF	0x0BF	'/'	BF

The ASCII values and hex codes listed represent the key to which these scan codes are assigned if in fact the machine was setup for the US keyboard driver. For machines with other keyboards (and country codes) these ASCII values may or may not be the same.

So what this all means is that because Zeus binds keys based on key location (ie scan code) and not ASCII characters in some cases you may have to remap the Zeus keyboard to suit the country configuration of your machine. For example with the Brief keymap loaded and the keyboard set up for US, the VK_0xBD is mapped to the FileClose and is represented by the ASCII Ctrl + '-' keys. On some non US machines this may in fact map to the ASCII Ctrl + '$' keys so some remapping will be required to return the key to the required Ctrl + '-' keys.

For exactly the same reason this may also cause some problems for QWERTY keyboards (as found on portable machines) and again some rebinding of keys may be required.

Printable Function List

Backspace

Moves the cursor back one position deleting the previous character

BackspaceEx

Moves the cursor back one position deleting the previous character and if you reach the start of the current line move the cursor to the end of the previous line

BackspaceSmart

Perform a backspace to a column position using the line above as a tab stop template.

BookMarkDrop0

Save the current line number against bookmark number 0

BookMarkDrop1

Save the current line number against bookmark number 1

BookMarkDrop2

Save the current line number against bookmark number 2

BookMarkDrop3

Save the current line number against bookmark number 3

BookMarkDrop4

Save the current line number against bookmark number 4

BookMarkDrop5

Save the current line number against bookmark number 5

BookMarkDrop6

Save the current line number against bookmark number 6

BookMarkDrop7

Save the current line number against bookmark number 7

BookMarkDrop8

Save the current line number against bookmark number 8

BookMarkDrop9

Save the current line number against bookmark number 9

BookMarkGoto

Move the cursor to the line stored against the bookmark selected. A valid bookmark is any number between 0 and 9.

BookMarkGoto0

Move the cursor to the line stored against the bookmark number 0

BookMarkGoto1

Move the cursor to the line stored against the bookmark number 1

BookMarkGoto2

Move the cursor to the line stored against the bookmark number 2

BookMarkGoto3

Move the cursor to the line stored against the bookmark number 3

BookMarkGoto4

Move the cursor to the line stored against the bookmark number 4

BookMarkGoto5

Move the cursor to the line stored against the bookmark number 5

BookMarkGoto6

Move the cursor to the line stored against the bookmark number 6

BookMarkGoto7

Move the cursor to the line stored against the bookmark number 7

BookMarkGoto8

Move the cursor to the line stored against the bookmark number 8

BookMarkGoto9

Move the cursor to the line stored against the bookmark number 9

BraceMatch

Finds the matching brace for the brace character at the current cursor position. A valid brace character is one of the following characters:

"[]{}<>()"

BraceMatchForward

Finds the matching reverse brace for the forward brace character at the current cursor position. A valid forward brace character is one of the following characters:

"[{<("

BraceMatchForwardEx

Search for the next brace charcater in a forward direction.

BraceMatchReverse

Finds the matching forward brace for the reverse brace character at the current cursor position. A valid reverse brace character is one of the following characters:

"]}>)"

BraceMatchReverseEx

Search for the next brace charcater in a reverse direction.

CharCopyFromLineAbove

Copies a character to the current line based on the corresponding character at the same cursor position in the line above.

CharCopyFromLineBelow

Copies a character to the current line based on the corresponding character at the same cursor position in the line below

CharDelete

Deletes the character at the current cursor position

CharQuote

Causes the next character to be enter to be treated literally even if it is a command keystroke

CharSwapNext

Swap the current character with the next character

CharSwapPrevious

Swap the current character with the previous character

ClipboardPaste

Paste the contents of the clipboard into the current cursor position

ClipboardPasteAndMark

Paste the contents of the clipboard into the current cursor position and mark the text that was just added

CompilerCompile

Compiles the currently active document

CompilerOutputCopy

Copy the contents of the Compiler Output Window to the clipboard

CompilerOutputNext

Moves the to the next compiler output or warning, be it in the compiled document or in the output window itself

CompilerOutputPrevious

Moves the to the previous compiler output or warning, be it in the compiled document or in the output window itself

CompilerOutputView

Display the Compiler Output Window

CompilerSetup

Display the compiler setup dialog

Enter

Causes a new line to be enter with the cursor moving to the new line

EnterLine

Causes a new line to be enter with the cursor moving to the new line but do not do any smart indenting or smart brace processing

EnterNext

Causes a new line to be enter without splitting the current line, with the cursor moving to the new line

EnterOpen

Causes a the current line to be split but the cursor position is maintained at its current location

FileAnsiToOem

Translates the current document into the OEM-defined character set.

FileBackupReset

Turns the automatic file backup option off

FileBackupSet

Turns the automatic file backup option on

FileBackupToggle

Toggles the automatic file backup option on and off

FileBackupWrite

Forces a backup write for the current active document

FileClose

Close the currently active document

FileExit

Close the application, checking that all documents have been saved.

FileInsertAtCursor

Insert a file into the current active document at the current line number.

FileListAssociates

Display a list of all the files that share the same base file name as the currently active document. Any associated file can then be loaded by just selecting the required file from this list.

FileListDisplay

Display a list of all the currently active document and output windows.

FileListDisplayEx

Display a list of all the currently active document and output windows but use an improved dialog box.

FileName

Display the name of the currently active document.

FileNew

Create a new untitled document, making it the currently active document.

FileOemToAnsi

Translates the current document from the OEM-defined character set into either an ANSI or a wide-character string.

FileOpen

Open a file from disk, making it the currently active document.

FileOpenEx

Open a file from disk, making it the currently active document but use the directory of the currently active file as the starting directory of the open dialog.

FileOpenInLine

Try to open the file as described by the text of the current line. This can be used to open include files.

FilePrint

Print the current active document.

FilePrintAll

Print all the currently open documents.

FilePrintSetup

Display the Print Setup dialog, thus allowing the printer to be configured.

FileReadOEMReset

Turns the File Read as OEM option off.

FileReadOEMSet

Turns the File Read as OEM option on.

FileReadOEMToggle

Toggles the File Read as OEM option on and off.

FileReadOnlyModeReset

Reset the file open read only mode. The read only mode determines the mode all subsequent files are opened.

FileReadOnlyModeSet

Set the file open read only mode. The read only mode determines the mode all subsequent files are opened.

FileReadOnlyModeToggle

Toggles the file open in read only mode. The read only mode determines the mode all subsequent files are opened.

FileReadOnlyReset

Reset the read only status of the current file. If the read only mode is set the file cannot be modified.

FileReadOnlySet

Set the read only status of the current file. If the read only mode is set the file cannot be modified.

FileReadOnlyToggle

Toggles the read only status of the current file. If the read only mode is set the file cannot be modified.

FileReload1

Reload the most recently open document number 1.

FileReload2

Reload the most recently open document number 2.

FileReload3

Reload the most recently open document number 3.

FileReload4

Reload the most recently open document number 4.

FileReload5

Reload the most recently open document number 5.

FileReload6

Reload the most recently open document number 6.

FileReload7

Reload the most recently open document number 7.

FileReload8

Reload the most recently open document number 8.

FileReload9

Reload the most recently open document number 9.

FileReloadCurrent

Reload the currently active document from disk.

FileSave

Save the currently active document.

FileSaveAll

Save all the currently active documents.

FileSaveAllNamed

Save all the currently active documents that are named. This means untitled documents will not be saved.

FileSaveAs

Save the currently active document under a different name.

FileTouch

Force's the time stamp of the currently active file to be touched.

FileWriteOEMReset

Turns the File Write as OEM option off.

FileWriteOEMSet

Turns the File Write as OEM option on.

FileWriteOEMToggle

Toggles the File Write as OEM option on and off.

FunctionFindAll

Find all the function definitions for the currently active document.

FunctionFindNext

Find the next function definition for the currently active document.

FunctionFindPrevious

Find the previous function definition for the currently active document.

HelpAbout

Display the help about dialog box.

HelpIndex

Display the online help index information.

HelpKeys

Display the online help on keys information.

HelpQuickHelp

Perform a quick help search on the current word the currently selected text. This will perform a quick help keyword search of the online help files installed.

HelpQuickSearch

Display the Quick Search dialog. This will allow you to perform a quick help keyword search of the online help files installed.

HelpQuickSetup

Display the Quick Configuration dialog. This will allow you to install online help files that are to be searched by the quick help search engine.

HelpRegister

Display the shareware registration information.

HelpSupport

Display information on how to get technical support for this software.

HelpUsing

Display the help on using the online help facility.

InsertModeReset

Turns the character insert mode off.

InsertModeSet

Turns the character insert mode on.

InsertModeToggle

Toggles the character insert mode on and off.

InsertSpace

Insert a single space character but retain the cursor position.

IsEndOfLine

Returns true if the cursor is at the end of the current line.

IsOutsideLine

Returns true if the cursor is past the end of the current line.

IsStartOfLine

Returns true if the cursor is at the start of the current line.

IsWithinLine

Returns true if the cursor is within the current line.

LineCaseLower

Convert the current line to lower case.

LineCaseTranspose

Convert the current line upper case characters to lower case and vice versa.

LineCaseUpper

Convert the current line to upper case.

LineCopy

Copy the current line to the clipboard.

LineCopyAppend

Copy the current line and append the result to the clipboard.

LineCopyEnd

Copy the text from the current cursor to the end of the line to the clipboard.

LineCopyEndAppend

Copy from the current cursor to the end of the line and append the result to the clipboard.

LineCut

Cut the current line to the clipboard.

LineCutAppend

Cut the current line and append the result to the clipboard.

LineCutEnd

Cut the text from the current cursor to the end of the line to the clipboard.

LineCutEndAppend

Cut from the current cursor to the end of the line and append the result to the clipboard.

LineCutEndEx

Cut the text from the current cursor to the end of the line to the clipboard or join the line with the line below if the cursor is currently at the end of the line.

LineCutStart

Cut the text from the current cursor to the start of the line to the clipboard.

LineDelete

Delete the current line. The line is not added to the clipboard.

LineDeleteEnd

Delete all the characters in the current line starting form the current position up to the end of the line.

LineDeleteEndEx

Clear the text from the current cursor position to the end of the line or join the line with the line below if the cursor is currently at the end of the line.

LineDeleteStart

Delete all the characters in the current line between the current position and the start of the line.

LineGoto

Display the line goto dialog box. From here you can enter the line number to which the cursor should be moved.

LineJoinNext

Join the current line with the next line.

LineJoinPrevious

Join the current line with the previous line.

LineTextReverse

Reverse the order of all the text contained in the current line.

LineWrap

Perform a check for line wrap on the current line and if it is found to be too long it will be wrapped. In the case that the current line does get wrapped this function does not check to see if the resulting new line also needs to be wrapped.

LineWrapEx

Perform a check for line wrap on the current line and if it is found to be too long it will be wrapped. In the case that the current line does get wrapped this function also checks to see if the resulting new lines also need to be wrapped and if they do this wrapping will also be performed.LineWrapMarkedArea.

LineWrapReset

Turns the automatic line wrap feature off.

LineWrapSet

Turns the automatic line wrap feature on.

LineWrapToggle

Toggles the automatic line wrap feature on and off.

MacroExecuteScript

Display the macro execute dialog which allows you to load and execute a macro script file.

MacroExecute1

Run the currently install macro number 1. If no macro has been install this command has no effect.

MacroExecute2

Run the currently install macro number 2. If no macro has been install this command has no effect.

MacroExecute3

Run the currently install macro number 3. If no macro has been install this command has no effect.

MacroExecute4

Run the currently install macro number 4. If no macro has been install this command has no effect.

MacroExecute5

Run the currently install macro number 5. If no macro has been install this command has no effect.

MacroExecute6

Run the currently install macro number 6. If no macro has been install this command has no effect.

MacroExecute7

Run the currently install macro number 7. If no macro has been install this command has no effect.

MacroExecute8

Run the currently install macro number 8. If no macro has been install this command has no effect.

MacroExecute9

Run the currently install macro number 9. If no macro has been install this command has no effect.

MacroExecute10

Run the currently install macro number 10. If no macro has been install this command has no effect.

MacroExecute11

Run the currently install macro number 11. If no macro has been install this command has no effect.

MacroExecute12

Run the currently install macro number 12. If no macro has been install this command has no effect.

MacroExecute13

Run the currently install macro number 13. If no macro has been install this command has no effect.

MacroExecute14

Run the currently install macro number 14. If no macro has been install this command has no effect.

MacroExecute15

Run the currently install macro number 15. If no macro has been install this command has no effect.

MacroLoad

Load a macro from file, making it the current macro.

MacroPlay

Play the current macro, be it recorded or loaded from disk.

MacroRecordReset

Reset the macro recording state. When the macro recording state is set all keyboard keystrokes are recorded against the current macro.

MacroRecordSet

Set the macro recording state. When the macro recording state is set all keyboard keystrokes are recorded against the current macro.

MacroRecordToggle

Toggles the macro recording state. When the macro recording state is set all keyboard keystrokes are recorded against the current macro.

MacroRepeat

Display the macro repeat dialog so that the current macro can be played a repeated number of times.

MacroSave

Save the current macro to a macro file.

MarkBlockReset

Turns the block marking mode off.

MarkBlockSet

Turns the block marking mode on.

MarkBlockSetEx

Turns the block marking mode on but remove any previous mark that may have been active.

MarkBlockToggle

Toggles the block marking mode on and off.

MarkBlockToggleEx

Toggles the block marking mode on or off but do not remove any existing marked area.

MarkCaseLower

Convert the marked text to lower case.

MarkCaseTranspose

Convert the marked text lower case characters to upper case and vice versa.

MarkCaseUpper

Convert the marked text to upper case.

MarkColumnReset

Turns the column marking mode off.

MarkColumnSet

Turns the column marking mode on.

MarkColumnSetEx

Turns the column marking mode on but remove any previous mark that may have been active.

MarkColumnToggle

Toggles the column marking mode on and off.

MarkColumnToggleEx

Toggles the column marking mode on or off but do not remove any existing marked area.

MarkCopy

Copy the marked area to clipboard.

MarkCopyEx

Copy the marked area or the current line to clipboard.

MarkCopyToCursor

Copy the marked area to the current cursor location.

MarkCopyToCursorEx

Copy the marked area to the current cursor location but retain the marked area after the copy is complete.

MarkCursorEnd

Move the cursor to the end of the marked area.

MarkCursorStart

Move the cursor to the start of the marked area.

MarkCursorToggle

Toggles the cursor position between the beginning and end of the currently marked area.

MarkCut

Cut the marked area to clipboard.

MarkCutEx

Cut the marked area or the current line to clipboard.

MarkDelete

Delete the marked area. The text deleted is not added to the clipboard.

MarkDeleteEx

Delete the marked area or the current line. The text deleted is not added to the clipboard.

MarkFillWithChar

Not yet implemented.

MarkFillWithSpace

Fill the currently marked area with space characters. This function will add characters past the end of line if the line is so marked.

MarkFillWithSpaceEx

Fill the currently marked area with space characters. This function will not add characters past the end of line.

MarkHide

Remove the current marked area, leaving the marked text unchanged.

MarkLineReset

Turns the line marking mode on.

MarkLineSet

Turns the line marking mode off.

MarkLineSetEx

Turns the line marking mode on but does not remove any previous mark that may have been active.

MarkLineToggle

Toggles the line marking mode on and off.

MarkLineToggleEx

Toggles the line marking mode on or off but do not remove any existing marked area.

MarkMoveToCursor

Move the marked area to the current cursor location.

MarkMoveToCursorEx

Move the marked area to the current cursor location but retain the marked area after the move is complete.

MarkPaste

Paste the contents of the clipboard, replacing any text that has been marked.

MarkPasteEx

Paste the contents of the clipboard, replacing any text that has been marked, or just insert the clipboard data if no area has been marked (Brief like paste operation).

MarkPrint

Print the currently marked area.

MarkSelectAll

Mark the entire contents of the currently active document leaving the document in marking mode.

MarkSelectAllEx

Mark the entire contents of the currently active document but turn of the marking once the document text has been selected.

MarkShiftLeft

Shift the currently marked area one tab space the left.

MarkShiftLeftEx

Shift the currently marked area to the previous tab stop.

MarkShiftRight

Shift the currently marked area one tab space the right.

MarkShiftRightEx

Shift the currently marked area to the next tab stop.

MarkTextReverse

Not yet implemented.

MarkWordCurrent

Mark the word under the current cursor location.

MarkWordEnd

Mark from the current cursor location to the end of the current word.

MarkWordStart

Mark from the start of the current word up to the current cursor location.

MarkWriteToFile

Write the currently marked area to file.

MoveDocumentCenter

Move the cursor to the center of the currently active document.

MoveDocumentEnd

Move the cursor to the end of the currently active document.

MoveDocumentEndEx

Move the cursor of the currently active document catering for the special Brief navigation sequence, end of line, end of page and end of document.

MoveDocumentStart

Move the cursor to the start of the currently active document.

MoveDocumentStartEx

Move the cursor of the currently active document catering for the special Brief navigation sequence, start of line, start of page and start of document.

MoveLineCenter

Move the cursor to the center of the current line.

MoveLineDown

Move the cursor down one line position.

MoveLineDownAndFirst

Move the cursor down to the next line and position the cursor at the first non-white space character of that line.

MoveLineDownAndLast

Move the cursor down to the next line and position the cursor at the first non-white space character of that line.

MoveLineEnd

Move the cursor to the end of the line.

MoveLineFirst

Move the cursor to the first non-white character in the current line.

MoveLineHome

Move the cursor to the start of the line.

MoveLineLeft

Move the cursor one character position to the left.

MoveLineLeftEx

Move the cursor one character position to the left but reposition the cursor to the end of the pervious

line if you reach the start of the current line.

MoveLineLeftEdge

Move the cursor to the character at the left edge of the screen.

MoveLineRight

Move the cursor one character position to the right.

MoveLineRightEx

Move the cursor one character position to the right but reposition the cursor to the start of the next line you reach the end of the current line.

MoveLineRightEdge

Move the cursor to the character at the right edge of the screen.

MoveLineUp

Move the cursor up one line position.

MoveLineUpAndFirst

Move the cursor up to the previous line and position the cursor at the first non-white space character of that line.

MoveLineUpAndLast

Move the cursor up to the previous line and position the cursor at the last non-white space character of that line.

MovePageCenter

Move the cursor to the center of the current page.

MovePageDown

Move the cursor down one page.

MovePageEnd

Move the cursor to the end of the current page.

MovePageLeft

Move the cursor to the left one page.

MovePageRight

Move the cursor to the right one page.

MovePageStart

Move the cursor to the start of the current page.

MovePageUp

Move the cursor up one page.

MoveWordCenter

Move the cursor to the center of the current word.

MoveWordEnd

Move the cursor to the end of the current word.

MoveWordNext

Move the cursor to the start of the next word.

MoveWordNextEx

Move to the next word and continue onto the next line if you reach the end of the current line.

MoveWordPrevious

Move the cursor to the start of the previous word.

MoveWordPreviousEx

Move to the previous word and continue onto the previous line if you reach the beginning of the current line.

MoveWordStart

Move the cursor to the start of the current word.

NotSupported

Display a message saying 'This keystroke is not supported'.

OptionsColors

Display the Color Options dialog.

OptionsEditor

Display the Editor Options dialog.

OptionsElectric

Display the Templates Options dialog.

OptionsExtension

Display the Extensions Options dialog.

OptionsFilters

Display the Filter Options dialog.

OptionsFont

Display the Font dialog.

OptionsKeyboard

Display the Keyboard Options dialog.

OptionsMacros

Display the Macro Options dialog.

OptionsSpelling

Display the Spelling Options dialog box.

OptionsTools

Display the Tool Options dialog.

PopupMenuDisplay

Display the popup edit menu list.

ProjectClose

Close the currently open project.

ProjectCurrent

Display the name of the currently open project.

ProjectDisplay

Display the file that make up the currently open project.

ProjectExecute

Execute the program command line as defined in the projects settings dialog.

ProjectExecuteDebug

Execute the debug command line as defined in the projects settings dialog.

ProjectMake

Make the currently open project.

ProjectMakeAll

Make all the components of the currently open project.

ProjectOpen

Open a project file.

ProjectOutputCopy

Copy the contents of the Project Output Window to the clipboard.

ProjectOutputNext

Display the next entry from the project output window.

ProjectOutputPrevious

Display the previous entry from the standard output window.

ProjectOutputView

Display the Project Output Window.

ProjectSetup

Display the Project Options dialog.

Redo

Redo the last undo command.

ReplaceDialog

Display the Replace dialog.

ReplaceAllSet

Sets the "replace all" check box..

ReplaceAllReset

Resets the "replace all" check box.

ReplaceAllToggle

Toggles the "replace all" check box on or off.

ReplaceForward

Display the Replace dialog with the search direction set to forward.

ReplaceNext

Replace the next instance the search text as defined by the last run replace command.

ReplacePrevious

Replace the previous instance the search text as defined by the last run replace command.

ReplaceReverse

Display the Replace dialog with the search direction set to reverse.

ReplaceWordCurrent

Display the Replace dialog with the search text set to match the current word provide the cursor is over a valid word.

ScreenUpdate

Force a screen update in case the screen needs repainting (does not update commenting).

ScrollBarViewReset

Set the visible state of the document scroll bars to hidden.

ScrollBarViewSet

Set the visible state of the document scroll bars to showing.

ScrollBarViewToggle

Toggle the visible state of the document scroll bars.

ScrollLineDown

Scroll the currently active document down one line, maintaining the cursor at the same relative screen position.

ScrollLineDownEx

Scroll the currently active document down one line but do not try to maintain the cursor at the same relative screen position.

ScrollLineLeft

Scroll the currently active document left one character position.

ScrollLinePageCenter

Scroll the current line to the center of the page.

ScrollLinePageEnd

Scroll the current line to the end of the page.

ScrollLinePageStart

Scroll the current line to the start of the page.

ScrollLineRight

Scroll the currently active document right one character position.

ScrollLineUp

Scroll the currently active document up one line, maintaining the cursor at the same relative screen position.

ScrollLineUpEx

Scroll the currently active document up one line but do not try to maintain the cursor at the same relative screen position.

ScrollLockReset

Turns the scroll locking feature off.

ScrollLockSet

Turns the scroll locking feature on.

ScrollLockToggle

Toggles the scroll locking feature on and off.

SearchCaseReset

Turns the search case sensitivity off.

SearchCaseSet

Turns the search case sensitivity on.

SearchCaseToggle

Toggles the search case sensitivity on and off.

SearchDialog

Display the search dialog.

SearchDialogEx

Display the search dialog but will automatically dismiss the dialog after the find key is pressed.

SearchDirectionReset

Turns the search direction to down.

SearchDirectionSet

Turns the search direction to up.

SearchDirectionToggle

Toggles the search direction between forward and reverse.

SearchForward

Display the search dialog with the search direction set to forward.

SearchForwardCount

Not yet implemented.

SearchNext

Repeat the last search in the forward direction.

SearchPrevious

Repeat the last search in the reverse direction.

SearchRegexpReset

Turns the search with regular expression off.

SearchRegexpSet

Turns the search with regular expression on.

SearchRegexpToggle

Toggles the search with regular expression on and off.

SearchReverse

Display the search dialog with the search direction set to reverse.

SearchReverseCount

Not yet implemented.

SearchWordCurrent

Display the Search dialog with the search text set to match the current word provide the cursor is over a valid word.

SearchWordCurrentNext

Search for the next occurrence of the current word without displaying the search dialog.

SearchWordCurrentPrevious

Search for the previous occurrence of the current word without displaying the search dialog.

SearchWordReset

Turns the search whole word option off.

SearchWordSet

Turns the search whole word option on.

SearchWordToggle

Toggles the search whole word option on and off.

SortAscending

Sort the file in ascending order.

SortDescending

Sort the file in descending order.

SortMarkedAscending

Sort the marked lines in ascending alphabetical order. Note that this function will alway sorts the entire line but also note that the actual sorting keys are defined by the marked area. This means that for example if a portion of a line is marked with column marking the range of lines marked will be sorted using the text select which is not necessarily the entire text of that line.

SortMarkedDescending

Sort the marked lines in descending alphabetical order. Note that this function will alway sorts the entire line but also note that the actual sorting keys are defined by the marked area. This means that for example if a portion of a line is marked with column marking the range of lines marked will be sorted using the text select which is not necessarily the entire text of that line.

SoundError

Produce the error sound.

SoundNote

Produce the note sound.

SoundOk

Produce the OK sound.

SoundQuestion

Produce the question sound.

SoundWarning

Produce the warning sound.

SpaceDelete

Not yet implemented.

SpaceDeleteEnd

Not yet implemented.

SpaceDeleteStart

Not yet implemented.

SpellingDocument

Check the currently active document for spelling errors.

SpellingWordCurrent

Check the current word for spelling errors.

StampDay

Insert the numerical day value into the currently active document.

StampMonth

Insert the numerical month value into the currently active document.

StampYear

Insert the numerical year value into the currently active document.

StampDateTime

Insert the full date/time stamp value into the currently active document.

StampTime

Insert the time stamp value into the currently active document.

StampFileName

Insert the name of current file into the currently active document.

StampFileSize

Insert the size of current file into the currently active document.

StampFileDateTime

Insert the date/time stamp when file was last modified into the currently active document.

StandardOutputCopy

Copy the contents of the Standard Output Window to the clipboard.

StandardOutputNext

Display the next entry from the standard output window.

StandardOutputPrevious

Display the previous entry from the standard output window.

StandardOutputView

Display the Standard Output Window.

StatusBarViewReset

Reset the status bar display state. When the display state is set the status bar is visible.

StatusBarViewSet

Set the status bar display state. When the display state is set the status bar is visible.

StatusBarViewToggle

Toggles the status bar display state. When the display state is set the status bar is visible.

Tab

If there is text that has been marked, move the marked text to the right by one tab character, else just insert a tab character.

TabBack

If there is text that has been marked, move the marked text to the left by one tab character, else just move the cursor back one tab stop.

TabBackChar

Move the cursor back one tab stop.

TabBackSmart

Perform a back tab to a column position using the line above as a tab stop template.

TabChar

Insert a tab character.

TabHard

Always insert a tab character irrespective of the current tabs as spaces editor option.

TabSmart

Perform a tab to a column position using the line above as a tab stop template.

TabSoft

Always insert a tab character as spaces irrespective of the current tabs as spaces editor option.

TagsSetup

Display the tag setup dialog.

TagsBuild

Build the tag file using the build tag command.

TagsDisplayNext

Display the next tag found.

TagsDisplayPrevious

Display the previous tag found.

TagsDisplayRedo

Redo the last tag display undo operation.

TagsDisplayResults

Display the all the matching tags found.

TagsDisplayTagFile

Display the contents of the tag file.

TagsDisplayUndo

Undo the last tag display operation.

TagsFindDialog

Display the Search Tag dialog.

TagsFindWordCurrent

Find the current word in the tag file.

TagsFindWordCurrent1

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsFindWordCurrent2

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsFindWordCurrent3

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsFindWordCurrent4

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

ToolBarViewReset

Reset the tool bar display state. When the display state is set the tool bar is visible.

ToolBarViewSet

Set the tool bar display state. When the display state is set the tool bar is visible.

ToolBarViewToggle

Toggles the tool bar display state. When the display state is set the tool bar is visible.

ToolsCommand

Display the DOS Command Line dialog.

ToolsExecute1

Run the currently install tool number 1. If no tool has been install this command has no effect.

ToolsExecute2

Run the currently install tool number 2. If no tool has been install this command has no effect.

ToolsExecute3

Run the currently install tool number 3. If no tool has been install this command has no effect.

ToolsExecute4

Run the currently install tool number 4. If no tool has been install this command has no effect.

ToolsExecute5

Run the currently install tool number 5. If no tool has been install this command has no effect.

ToolsExecute6

Run the currently install tool number 6. If no tool has been install this command has no effect.

ToolsExecute7

Run the currently install tool number 7. If no tool has been install this command has no effect.

ToolsExecute8

Run the currently install tool number 8. If no tool has been install this command has no effect.

ToolsExecute9

Run the currently install tool number 9. If no tool has been install this command has no effect.

ToolsExecute10

Run the currently install tool number 10. If no tool has been install this command has no effect.

ToolsExecute11

Run the currently install tool number 11. If no tool has been install this command has no effect.

ToolsExecute12

Run the currently install tool number 12. If no tool has been install this command has no effect.

ToolsExecute13

Run the currently install tool number 13. If no tool has been install this command has no effect.

ToolsExecute14

Run the currently install tool number 14. If no tool has been install this command has no effect.

ToolsExecute15

Run the currently install tool number 15. If no tool has been install this command has no effect.

ToolsShell

Spawn a DOS command line session.

Undo

Undo the last made change.

UndoFlush

Flush the undo buffer for the currently active document.

Version

Display the current version of the software.

WindowArrange

Arrange the currently active MDI windows.

WindowCascade

Cascade the currently active MDI windows.

WindowCloseAll

Close all the currently active MDI windows, making sure all changes have been saved.

WindowMaximizeAll

Maximize all the currently open windows.

WindowMinimizeAll

Minimize all the currently open windows.

WindowNext

Move to the next active MDI window.

WindowPrevious

Move to the previous active MDI window.

WindowRestoreAll

Restore the size of all the currently open windows.

WindowTile

Tile the currently active MDI windows.

WindowTileHorizontal

Tile the currently active MDI windows with a horizontal aspect.

WindowTileVertical

Tile the currently active MDI windows with a vertical aspect.

WordCaseLower

Convert the current word to lower case.

WordCaseTranspose

Convert the current word lower case characters to upper case and vice versa.

WordCaseUpper

Convert the current word to upper case.

WordCopy

Copy the current word to clipboard.

WordCut

Cut the current word to clipboard.

WordDelete

Delete the current word.

WordDeleteEnd

Delete from the current position to the end of the current word.

WordDeleteNext

Delete the next word.

WordDeleteNextEx

Delete the next word or join the line with the next line if there is no next word to delete.

WordDeletePrevious

Delete the previous word.

WordDeletePreviousEx

Delete the previous word or join the line with the previous line if there is no previous word to delete.

WordDeleteStart

Delete from the start of the word up to the current position.

WordTextReverse

Perform a text reversal operation on the current word.

Functions Not Yet Done

Some of the Zeus functions can be bound but their functionality is not yet supplied. The list of the functions that have not yet been defined are shown below.

FilePrintAll
LineTextReverse
MarkMoveToCursor
MarkCopyToCursor
MarkTextReverse
ProjectExecute
ProjectMakeAll
ProjectDisplay
SearchForwardCount
SearchReverseCount
SpaceDelete
SpaceDeleteEnd
SpaceDeleteStart

Function List

Below is a list of the editing functions provided by Zeus and a description of what each function does. For a version of this list that can be sent to the printer see the printable function list provided. Please note that a very small number of these functions have been added for planned future enhancements, but as yet their functionality has not been implemented. To find out more about a particular function just click in the function name.

FileAnsiToOem
FileBackupReset
FileBackupSet
FileBackupToggle
FileBackupWrite
FileClose
FileExit
FileInsertAtCursor
FileListAssociates
FileListDisplay
FileListDisplayEx
FileName
FileNew
FileOemToAnsi
FileOpen
FileOpenEx
FileOpenInLine
FilePrint
FilePrintAll
FilePrintSetup
FileReadOEMReset
FileReadOEMSet
FileReadOEMToggle
FileReadOnlyModeReset
FileReadOnlyModeSet
FileReadOnlyModeToggle
FileReadOnlyReset
FileReadOnlySet
FileReadOnlyToggle
FileReload1
FileReload2
FileReload3
FileReload4
FileReload5
FileReload6
FileReload7
FileReload8
FileReload9
FileReloadCurrent
FileSave
FileSaveAll
FileSaveAllNamed
FileSaveAs
FileTouch
FileWriteOEMReset
FileWriteOEMSet
FileWriteOEMToggle
FunctionFindAll
FunctionFindNext
FunctionFindPrevious
HelpAbout
HelpIndex
HelpKeys
HelpQuickHelp
HelpQuickSearch

HelpQuickSetup
HelpRegister
HelpSupport
HelpUsing
InsertModeReset
InsertModeSet
InsertModeToggle
InsertSpace
IsEndOfLine
IsOutsideLine
IsStartOfLine
IsWithinLine
LineCaseLower
LineCaseTranspose
LineCaseUpper
LineCopy
LineCopyAppend
LineCopyEnd
LineCopyEndAppend
LineCut
LineCutAppend
LineCutEnd
LineCutEndAppend
LineCutEndEx
LineCutStart
LineDelete
LineDeleteEndEx
LineDeleteEnd
LineDeleteStart
LineGoto
LineJoinNext
LineJoinPrevious
LineTextReverse
LineWrap
LineWrapEx
LineWrapMarkedArea
LineWrapReset
LineWrapSet
LineWrapToggle
MacroExecuteScript
MacroExecute1
MacroExecute2
MacroExecute3
MacroExecute4
MacroExecute5
MacroExecute6
MacroExecute7
MacroExecute8
MacroExecute9
MacroExecute10
MacroExecute11
MacroExecute12
MacroExecute13
MacroExecute14
MacroExecute15

MacroLoad
MacroPlay
MacroRecordReset
MacroRecordSet
MacroRecordToggle
MacroRepeat
MacroSave
MarkBlockReset
MarkBlockSet
MarkBlockSetEx
MarkBlockToggle
MarkBlockToggleEx
MarkCaseLower
MarkCaseTranspose
MarkCaseUpper
MarkColumnReset
MarkColumnSet
MarkColumnSetEx
MarkColumnToggle
MarkColumnToggleEx
MarkCopy
MarkCopyEx
MarkCopyToCursor
MarkCopyToCursorEx
MarkCursorEnd
MarkCursorStart
MarkCursorToggle
MarkCut
MarkCutEx
MarkDelete
MarkDeleteEx
MarkFillWithChar
MarkFillWithSpace
MarkFillWithSpaceEx
MarkHide
MarkLineReset
MarkLineSet
MarkLineSetEx
MarkLineToggle
MarkLineToggleEx
MarkMoveToCursor
MarkMoveToCursorEx
MarkPaste
MarkPasteEx
MarkPrint
MarkSelectAll
MarkSelectAllEx
MarkShiftLeft
MarkShiftLeftEx
MarkShiftRight
MarkShiftRightEx
MarkTextReverse
MarkWordCurrent
MarkWordEnd
MarkWordStart

MarkWriteToFile
MoveDocumentCenter
MoveDocumentEnd
MoveDocumentEndEx
MoveDocumentStart
MoveDocumentStartEx
MoveLineCenter
MoveLineDown
MoveLineDownAndFirst
MoveLineDownAndLast
MoveLineEnd
MoveLineFirst
MoveLineHome
MoveLineLeft
MoveLineLeftEx
MoveLineLeftEdge
MoveLineRight
MoveLineRightEx
MoveLineRightEdge
MoveLineUp
MoveLineUpAndFirst
MoveLineUpAndLast
MovePageCenter
MovePageDown
MovePageEnd
MovePageLeft
MovePageRight
MovePageStart
MovePageUp
MoveWordCenter
MoveWordEnd
MoveWordNext
MoveWordNextEx
MoveWordPrevious
MoveWordPreviousEx
MoveWordStart
NotSupported
OptionsColors
OptionsEditor
OptionsElectric
OptionsExtension
OptionsFilters
OptionsFont
OptionsKeyboard
OptionsMacros
OptionsSpelling
OptionsTools
PopupMenuDisplay
ProjectClose
ProjectCurrent
ProjectDisplay
ProjectExecute
ProjectExecuteDebug
ProjectMake
ProjectMakeAll

ProjectOpen
ProjectOutputCopy
ProjectOutputNext
ProjectOutputPrevious
ProjectOutputView
ProjectSetup
Redo
ReplaceDialog
ReplaceForward
ReplaceNext
ReplacePrevious
ReplaceReverse
ReplaceWordCurrent
ScreenUpdate
ScrollBarViewReset
ScrollBarViewSet
ScrollBarViewToggle
ScrollLineDown
ScrollLineDownEx
ScrollLineLeft
ScrollLinePageCenter
ScrollLinePageEnd
ScrollLinePageStart
ScrollLineRight
ScrollLineUp
ScrollLineUpEx
ScrollLockReset
ScrollLockSet
ScrollLockToggle
SearchCaseReset
SearchCaseSet
SearchCaseToggle
SearchDialog
SearchDialogEx
SearchDirectionReset
SearchDirectionSet
SearchDirectionToggle
SearchForward
SearchForwardCount
SearchNext
SearchPrevious
SearchRegexpReset
SearchRegexpSet
SearchRegexpToggle
SearchReverse
SearchReverseCount
SearchWordCurrent
SearchWordCurrentNext
SearchWordCurrentPrevious
SearchWordReset
SearchWordSet
SearchWordToggle
SortAscending
SortDescending
SortMarkedAscending

SortMarkedDescending
SoundError
SoundNote
SoundOk
SoundQuestion
SoundWarning
SpaceDelete
SpaceDeleteEnd
SpaceDeleteStart
SpellingDocument
SpellingWordCurrent
StampDateTime
StampDay
StampFileName
StampFileSize
StampFileDateTime
StampMonth
StampTime
StampYear
StandardOutputCopy
StandardOutputNext
StandardOutputPrevious
StandardOutputView
StatusBarViewReset
StatusBarViewSet
StatusBarViewToggle
Tab
TabBack
TabBackSmart
TabBackChar
TabChar
TabHard
TabSoft
TabSmart
TagsBuildTagsDisplayNextTagsDisplayPreviousTagsDisplayRedoTagsDisplayResultsTagsDisplayTagFileTagsDisplayUndoTagsFindDialogIDH_TagsFindDialog
TagsFindWordCurrentTagsFindWordCurrent1TagsFindWordCurrent2TagsFindWordCurrent3TagsFindWordCurrent4TagsSetupToolBarViewResetToolBarViewSet
ToolBarViewToggle
ToolsCommand
ToolsExecute1
ToolsExecute2
ToolsExecute3
ToolsExecute4
ToolsExecute5
ToolsExecute6
ToolsExecute7
ToolsExecute8
ToolsExecute9
ToolsExecute10
ToolsExecute11
ToolsExecute12
ToolsExecute13
ToolsExecute14
ToolsExecute15

ToolsShell
Undo
UndoFlush
Version
WindowArrange
WindowCascade
WindowCloseAll
WindowMaximizeAll
WindowMinimizeAll
WindowNext
WindowPrevious
WindowRestoreAll
WindowTile
WindowTileHorizontal
WindowTileVertical
WordCaseLower
WordCaseTranspose
WordCaseUpper
WordCopy
WordCut
WordDelete
WordDeleteEnd
WordDeleteNext
WordDeleteNextEx
WordDeletePrevious
WordDeletePreviousEx
WordDeleteStart
WordTextReverse

Menu Commands

In addition to providing a comprehensive keyboard interface Zeus also provides a menu interface, which mirrors some of the more common keyboard functions. Below is a list of the groups of menu functions provided by Zeus. For more information on any of these groups just select the required item from the list.

File Menu

The File menu provides commands for creating new files, managing existing files, printing files and exiting the application. It also provides a mechanism for loading files by selecting them from the list of most recently used files. Below is a list of the items that make up the File menu. For more information on any of these items just select the required item from the list.

New Command

Open Command

Close Command

Save Command

Save As Command

Save All Command

Print Command

Print Setup Command

Reload Document Command

Exit Command

Edit Menu

The Edit menu provides commands that help to modify, search, navigate and manipulate the text in the currently active document. Below is a list of the items that make up the Edit menu. For more information on any of these items just select the required item from the list.

Undo Command

Redo Command

Cut Command

Copy Command

Paste Command

Clear Command

Select All Command

Find Command

Replace Command

Find Next Command

Find Pervious Command

Find Matching Brace Command

Goto Line Command

Goto Bookmark Command

View Menu

The View menu provides commands to change the visible state of the status bar and tool bar, to display the Project, Standard and Compiler output windows and to navigate the error message list. Below is a list of the items that make up the View menu. For more information on any of these items just select the required item from the list.

Toolbar Command

Status Command

Function List Command

Next Function Command

Previous Function Command

Document List Command

Project Output Command

Standard Output Command

Compiler Output Command

Next Error Command

Previous Error Command

Options Menu

The Options menu provides commands that allow you to configure every aspect of the Zeus editor. Using these commands it is possible to configure the visual aspects of the editor, special editor settings, compiler and project support, productivity aids, tools and macros. Below is a list of the items that make up the Options menu. For more information on any of these items just select the required item from the list.

Tools Command

Macros Command

Project Command

Compiler Command

Font Command

Colors Command

Editor Command

Filters Command

Keyboard Command

Templates Command

Extensions Command

Macros Menu

The Macro menu provides commands for creating, playing, loading and the saving of keyboard macros. You can also add macros to this menu using the Macro Options dialog. Below is a list of the items that make up the Macros menu. For more information on any of these items just select the required item from the list.

Execute Script

Load Command

Save Command

Record Command

Playback Command

Repeated Playback Command

Run Macro Command

Tags Menu

The Tags menu provides commands for performing tag searching. To use this menu the tag system needs to have been correctly configured using the Tags Options dialog. Below is a list of the items that make up the Tags menu. For more information on any of these items just select the required item from the list.

Build Tags File

Build Display Tags File

Tag Search

Tag Current Word

Undo Tag

Redo Tag

Tools Menu

The Tool menu provides commands for running DOS commands, exiting out to a DOS shell or running previously configured third party tools. These tools can be added to the menu by using the Tool Options dialog. Below is a list of the items that make up the Tools menu. For more information on any of these items just select the required item from the list.

DOS Command Line Command

DOS Shell Command

Run Tool Command

Compiler Menu

The Compiler menu provides commands for compiling the currently active document. To use this menu the compiler needs to have been correctly configured using the Compiler Options dialog. Below is a list of the items that make up the Compiler menu. For more information on any of these items just select the required item from the list.

Compile Command

Project Menu

The Project menu provides commands for selecting, building and managing the task of building a project. To use this menu the project needs to have been correctly configured using the Project Options dialog. Below is a list of the items that make up the Project menu. For more information on any of these items just select the required item from the list.

Open Command
Close Command
Make Command
Execute Debug

Spelling Menu

The Spelling menu provides commands for controlling the built-in spelling engine. To use this menu the spelling engine needs to have been correctly configured using the Spelling Options dialog. Below is a list of the items that make up the Spelling menu. For more information on any of these items just select the required item from the list.

Document Command

Current Word Command

Window Menu

The Window menu contains commands for displaying, managing and navigating through the list of currently active MDI windows. Below is a list of the items that make up the Window Menu. For more information on any of these items just select the required item from the list.

Cascade Command

Tile Command

Tile Vertical Command

Tile Horizontal Command

Arrange Icons Command

Close All Command

Next Window Command

Previous Window Command

Help Menu

The Help menu provides access to online help via the Windows Help System. The help provided covers information on virtually all aspects of the Zeus editor and its functionality. Also provided is a help search system that allows the user to quickly and easily access the information contained in any other third party Windows 3.x help file. Below is a list of the items that make up the Help menu. For more information on any of these items just select the required item from the list.

Using Help Command

Help on Keys Command

Help Index Command

Quick Configuration Command

Quick Help Command

Quick Search Command

About Zeus Command

Popup Menu

The Popup menu provides quick access to a few of the common edit functions and is activated using the right mouse button. Below is a list of the items that make up the Popup menu. For more information on any of these items just select the required item from the list.

Upper Case Command

Lower Case Command

Toggle Case Command

Open Include Command

Upper Case Command (Popup menu)

The Upper Case command will convert the marked text to upper case. This command is only available if some text is marked.

Lower Case Command (Popup menu)

The Lower Case command will convert the marked text to lower case. This command is only available if some text is marked.

Toggle Case Command (Popup menu)

The Transpose Case convert the marked text lower case characters to upper case and vice versa. This command is only available if some text is marked.

Open Include Command (Popup menu)

The Open Include command will parse the current line for a valid file name. If a file name is found and attempt will be made to locate the file and load it for editing. A typical use for this command would be as a quick way of loading source include files etc.

New Command (File menu)

The New command creates MDI document window with the name of 'untitled.txt' and make it the currently active document.

Open Command (File menu)

The Open command displays the Open File dialog box from which a document (or multiple documents) can be selected for editing. To select more than one document use the standard Windows CUA multi-select commands.

Close Command (File menu)

The Close command will close the currently active document or view. In the case of documents if the document being closed has been modified but not saved a message box will be displayed asking if the changes should be saved, ignored or if the closed should be cancelled.

Save Command (File menu)

The Save command will save the currently active document to disk. If the document is 'untitled.txt' the Save As dialog will be displayed requesting that the user provide a name for the document to be saved. If you want to save all modified files and not just the file in the active document use the File | Save All command.

Save As Command (File menu)

The Save As command will display the Save File As dialog box which allows the user to save the currently active document to a document of a different name or to a different directory or to a different drive. If you enter the name of an existing document you will be asked if you want to overwrite the existing file.

Save All Command (File menu)

The Save All command will save the contents of all the currently loaded documents. In this sense it works just like the Save command except that it saves the contents of all documents and not just the currently active document.

Print Command (File menu)

The Print command will display the Print dialog box. This dialog allows you to select an appropriate printer to which the currently active document will be printed.

Print Setup Command (File menu)

The Printer Setup command will display the Print Setup dialog box. This dialog allows you to select the appropriate printer or configure the currently selected printer, prior to actually printing the currently active document.

Reload Document Command (File menu)

The reload document command is used a reload a previously loaded file which has been held in the history list of the most recently opened documents. Any document that is opened and then closed will automatically be added to the file history list. The list is displayed at the bottom of the File menu. To quickly re-open any of the most recently used documents just select the required item form the history list.

The one exception to this rule is that if the files are closed using the Windows | Close All or the MDI double click close then their names are not saved to the history list. This behaviour is by design as it provides the user with a method of controlling what files get added to the history list.

Exit Command (File menu)

The Exit command will close the application making sure that all the currently loaded documents have been saved. If there are any documents that have been modified but not saved you will be asked if the changes should be saved, ignored or if the request to exit should be cancelled.

Undo Command (Edit menu)

The Undo command lets you reverse the actions of the most recent edit commands. This means that any edit change that you make can be reversed, with the document returning to the state prior to the last edit. The levels of undo are only limited by the amount of memory held in the system and thus for all intensive purposes are unlimited. The undo buffer is flushed once a document is saved. This means that once you save the document you will not be able to undo the changes made prior to the save.

Line Wrap (Edit menu)

The Line Wrap command allows you to toggle the current line wrap option on or off. The current state of the line wrapping is also indicated in the status bar.

Redo Command (Edit menu)

The Redo command lets you reverse the actions of the most recent Undo command. The levels of undo and redo are only limited by the amount of memory held in the system and thus for all intensive purposes are unlimited. The redo buffer is flushed once a document is saved. This means that once you save the document you will not be able to redo the undo changes made prior to the save.

Cut Command (Edit menu)

The Cut command removes the marked text from the currently active document and places the text in the Clipboard. You can then choose Edit Paste to paste the text back into the document, into any other document, or into another application that supports the use of the clipboard. The text remains in the Clipboard so you can paste it as many times as you want.

Copy Command (Edit menu)

The Copy command copies the marked text from the currently active document and places the text in the Clipboard. You can then choose Edit Paste to paste the text back into the document, into any other document, or into another application that supports the use of clipboard. The text remains in the Clipboard so you can paste it as many times as you want.

In the case of the standard output, compiler output and project output windows, there is no need to mark any text as the copy command automatically copies the entire contents of these windows into the clipboard.

Paste Command (Edit menu)

The Paste command inserts any text currently held in the clipboard into the currently active document. The text is add to the document at the current cursor position and will insert or overwrite the existing text based on the current text insert mode. You can add data to the clipboard using the Edit Copy or Edit Cut commands, or alternatively, you can use any another application to add text to the clipboard provided it also supports the use of the clipboard.

Clear Command (Edit menu)

The Clear command deletes the marked text from the currently active document but the data is not added to the clipboard.

Select All Command (Edit menu)

The Select All command marks the entire contents of the currently active document.

Find Command (Edit menu)

The Find command will display the Find dialog box. This dialog allows you to search for a particular piece of text in the currently active document.

Replace Command (Edit menu)

The Replace command will display the Replace dialog box. This dialog allows you to search for and then replace a particular piece of text in the currently active document.

Find Next Command (Edit menu)

The Find Next command repeats the last Find command by searching the current document in an forward direction.

Find Pervious Command (Edit menu)

The Find Previous command repeats the last Find command by searching the current document in an reverse direction.

Find Matching Brace (Edit menu)

The Find Matching Brace command will locate the matching brace for the character located at the current cursor position. If the matching brace is found the cursor will be moved to the corresponding matching brace otherwise an error will be displayed in the status line. To return to the original cursor location just repeat the search for the matching brace. The following sets of matching braces are the supported:

"()" "[]" "{}", "[]", "<>"

Goto Line Command (Edit menu)

The Goto Line command will display the Goto Line dialog box. This dialog provides you with an easy way to navigate within the document using line numbers.

Goto Bookmark Command (Edit menu)

The Goto Bookmark command will display the Goto Bookmark dialog box. This dialog provides you with an easy way to navigate within the document using pre-defined bookmarks.

Bookmark Number

Enter the bookmark you wish to move to. The bookmark entered must be a number between 0 and 9 and the bookmark must already exist in the current document. Note that bookmarks are only local to the current document meaning you cannot jump to a bookmark in another none active document. Each document can have up to 10 bookmarks

Line Number

Enter the line number you wish to move to. If you specify a line greater than the last line of the document the Line Goto will take you to the last line of the file.

Goto Line Dialog

The Goto Line dialog lets you move the cursor to any particular line in the currently active document. For more information on each of the different sections of this dialog click on one of the items listed below.

Line Number

OK

Cancel

Help

Goto Bookmark Dialog

The Goto Bookmark dialog lets you move the cursor to a previously defined bookmark in the currently active document. The action of dropping the bookmark depends on the current active keyboard mapping so refer to the keyboard definition for more details.

Bookmark Number

OK

Cancel

Help

Function List Command (View menu)

The Function List command will display a list of all the function definitions in the currently active document. By selecting a function from the list Zeus will automatically position the cursor at the location of the function definition. The function searching algorithm used by Zeus can be configured using the Edit Extension dialog.

Next Function Command (View menu)

The Next Function command moves the cursor of the currently active document to the location of the next function definition.

Previous Function Command (View menu)

The Previous Function command moves the cursor of the currently active document to the location of the previous function definition.

Document List Command (View menu)

The Document List command displays a list of the currently loaded documents. You can select a document from the list and Zeus will make the selected document the currently active document.

Toolbar Command (View menu)

The Toolbar command can be used to show or hide the Toolbar control.

Status Bar Command (View menu)

The Status Bar command can be used to show or hide the Status Bar control.

Project Output Command (View menu)

The Project Output command will activate the project output MDI window. This window is used to display the error and warning messages generated during the make of the currently open project.

Standard Output Command (View menu)

The Standard Output command will activate the standard output MDI window. This window is used to display the output generated by any of the DOS Commands or the output of most recently used tool (provided it has been configured for output capture).

Compiler Output Command (View menu)

The Compiler Output command will activate the compiler output MDI window. This window is used to display the error and warning messages generated by the compilation of the currently active document.

Next Error Command (View menu)

The Next Error command moves the cursor of the currently active document to the location of the next error or warning message. This command is only available if there are error or warning messages in the compiler output window.

Previous Error Command (View menu)

The Previous Error command moves the cursor of the currently active document to the location of the previous error or warning message. This command is available if there are error or warning messages in the compiler output window.

Load Command (Macros menu)

The Load command will display the Load Macro dialog, which allows the user to select the macro to be loaded. Once loaded this macro becomes the currently active macro and can be played using the macro playback commands.

Save Command (Macros menu)

The Save command will display the Save Macro dialog, which allows the user to save the currently active macro. If the macro name supplied corresponds to an existing macro the user will be asked to confirm that existing macro should be overwritten.

Macro Execute Script Command (Macros menu)

The Macro Execute Script command will display the Macro Execute Script dialog, which allows the user to execute a predefined macro script. For more help on writing and running of macro scripts refer to the writing of macro scripts section of this help file.

Record Start/Stop Command (Macros menu)

The Record Start/Stop command starts or stops the macro recording process. The current record state of the macro is indicated on the status line. The recording process will replace the currently active macro with the newly recorded macro. Once the macro has been recorded it can be run using the macro playback commands or alternatively it can be saved to file as a more permanent form of recording.

Playback Command (Macros menu)

The Playback command will playback the currently active macro. This command will only work if a macro has been recorded or if an existing macro has been loaded from file.

Repeated Playback Command (Macros menu)

The Repeated Playback command allows the playback the currently active macro a multiple number of times. This command first displays the Repeated Playback dialog, which allows the user to specify the number of times the macro should be run. This command will only work if a macro has been recorded or if an existing macro has been loaded from file.

Run Macro Command (Macros menu)

The Run Macro command allows the user to run pre-defined macro file using the menu or keyboard interface. The macro file must have been previously created using the macro save command and the macro needs to have been configured to appear in the Macro menu list using the Macro Options dialog.

Macro Options Dialog

The Macro Options dialog allows the user to manage the macros appearing on the Macro Menu list. This dialog allows macros to be easily added, edited or removed from the menu list. For more information on each of the different sections of this dialog click on one of the items listed below.

- Find
- Add
- Update
- Delete
- Next
- Previous
- Name
- Menu Text
- Close
- Help

Note that as well as letting you bind keyboard macros this dialog also lets you bind Zeus macro script files to the macro menu. To bind a Zeus macro script file just select the file as you would a keyboard macro file.

Find

Displays the find macro dialog box which allows a macro file to be easily located form disk.

Add

Adds the macro to the list of installed macros provided all the required details have been supplied.

Update

Updates the selected macro with the modified details.

Delete

Deletes the selected macro from the list of installed macros.

Name

This is the name of the macro file. Use the Find function to help locate the macro file if necessary.

Menu Text

This is the text that is to be displayed in the macro menu item list.

Previous

Move the selected item to the previous location in the list.

Next

Move the selected item to the next available location in the list.

Close

Dismiss the dialog box.

DOS Command Line Command (Tools menu)

The DOS Command Line command presents the user with the DOS Command Line dialog box. This allows the user to run DOS commands like DIR, GREP or any other valid DOS command.

DOS Shell Command (Tools menu)

The DOS Shell command will present the user with a DOS command line session. To return to Zeus either switch back to the Zeus application from the task list or exit the shell session by typing exit at the command line prompt.

Run Tool Command (Tools menu)

The Run Tool command allows you to run pre-defined tool using a menu or keyboard interface. The tools must have been must have been first configured to appear in the Tools menu list using the Tools Options dialog box.

Open Command (Project menu)

The Open command displays the Project Open dialog to allow the user to select a project to be opened and the selected project then becomes the currently active project.

Close Command (Project menu)

The Close command allows the user to close the currently active project.

Execute Debug

The Execute Debug command allows the user to execute the debugger that has been configured for the currently open project. The debugger options should have been previously configured using the Project Options dialog.

Make Command (Project menu)

The Make command will build the currently open project. This command is only available if a project is currently open. The information on how to build the project should have been previously configured using the Project Options dialog.

Compile Command (Compiler menu)

The Compile command will compile the currently active document. The information on how to compile a document should have been previously configured using the Compile Options dialog .

Use Extra Diagnostics

If you are having trouble configuring a tool then use this option to provide extra debug information. By enabling this option Zeus will report a more verbose message if an error is detected.

Tool Options Dialog

The Tool Options dialog allows the user to manage the tools that make up the Tools Menu list. Tools can be easily add, edit or remove for the tools menu list. To use this dialog just supply the required information into the appropriate fields of the dialog box. For more information on each of the different sections of this dialog click on one of the items listed below.

- Find
- Add
- Update
- Delete
- Next
- Previous
- Use extra diagnostics
- Program Name
- Menu Text
- DOS Type
- Windows Type
- Command Line Type
- Working Directory
- Run Normal
- Run Hidden
- Run Minimized
- Run Maximized
- Arguments
- Ask for arguments
- Capture Standard Output
- Capture Standard Error
- Save document before running
- Close
- Help

If you have any problems getting the tool to run correctly refer to the Understanding the Tool Support section for more details. Also remember that the you can also include one of the many Zeus TAG macros in the Arguments entry field.

IMPORTANT NOTE: If you want to add an MS-DOS batch file to the tools menu and you want to capture the standard output produced by this batch file you will need to check the 'capture standard error' option. This will run the tool using the ZSTDERR.EXE instead of the MS-DOS COMMAND.COM and as such will fix the MS-DOS limitation of not being able to capture the standard output generated by a batch file. For this option to work correctly the Zeus install directory ***must be*** in the system path.

Find Tool

Displays the find tool dialog box.

Add

Adds the tool to the list of installed tools provided the required tool details have been supplied.

Update

Updates the selected tool with the modified tool details.

Delete

Delete the selected tool from the list of tools.

DOS Type

The tool listed is a DOS program.

Windows Type

The tool listed is a Windows program.

Command Line Type

The tool is run using the DOS COMMAND.COM program.

Program Name

The program name of the tool to be added. Use the Find function to help locate the tool if necessary. If the program name uses a long file name that contains a space character then you will need to enclose the name in quotes.

Example: "c:\a long file\test.exe"

Menu Text

The text that is to be displayed in the tools menu.

Arguments

The arguments to be used by the tool. This entry field represents the command line arguments portion of the command. If any of the arguments use a long file name that contains a space character then you will need to enclose the name in quotes.

Example: "c:\a long file\test.exe"

Work Directory

The directory in which the tool is to be run.

Run Normal

Run the tool in a normal or restored window.

Run Hidden

Run the tool in a hidden. If this option is used make sure the tool does not require any user input.

Run Minimized

Run the tool as a minimized icon on the desktop.

Run Maximized

Run the tool as a maximized window.

Capture Standard Output

Capture the standard output produced by the tool and display the results in the standard output window.

IMPORTANT NOTE: If you want to capture the standard output produced by an MS-DOS batch file you will need to also check the 'capture standard error' option. This will bypass the MS-DOS COMMAND.COM and will using the ZSTDERR.EXE instead and as such should fix the MS-DOS limitation of not being able to capture the standard output generated by a batch file. For this option to work the Zeus install directory ***must be*** in the system path.

Ask for arguments

Before running the tool ask the user to supply the tool with additional command line arguments.

Save document before running

This option will force the currently active document to be save before the tool is actually run.

Cascade Command (Window menu)

The Cascade command stacks all the currently open documents. This option arranges the document windows so that each is the same size as all others and the title of each document window is clearly visible.

Tile Command (Window menu)

The Tile command tiles all the currently open documents. This option arranges the document windows in such a way that none of the document windows are overlapped by any of the another document windows.

Tile Vertical Command (Window menu)

The Tile Vertical command tiles all the currently open documents with a vertical aspect. This option arranges the document windows in such a way that none of the document windows are overlapped by any of the another document windows with a preference for a vertical arrangement if possible.

Tile Horizontal Command (Window menu)

The Tile Horizontal command tiles all the currently open documents with a horizontal aspect. This option arranges the document windows in such a way that none of the document windows are overlapped by any of the another document windows with a preference for a horizontal arrangement if possible.

Arrange Icons Command (Window menu)

The Arrange Icons command rearranges any of the document windows that have been iconized. The arrangement will try to space the icons evenly across the bottom of the main application window.

Close All Command (Window menu)

The Close All command closes all the currently open document windows and all the currently open output windows. If any of the document windows contain text that has been modified the user will be asked if the changes should be saved, ignored or if the close should be cancelled. Note that unlike the single window close command, by using this method of closing the documents the most recently change list of documents is left unaffected.

Next Window Command (Window menu)

The Next Window command will activate the next document window or output window in the list of currently active MDI windows.

Previous Window Command (Window menu)

The Previous Window command will activate the previous document window or output window in the list of currently active MDI windows.

Activate Window Command (Window menu)

At the bottom of the Window menu is a list of all the currently active MDI windows. You can switch to any one of the another windows by just selecting it from the list.

Using Help Command (Help menu)

The Using Help command displays information on how to use the Windows online help system. Use this command to learn how to use this help file effectively.

Help On Keys Command (Help menu)

The Help on Keys command provides a quick method of accessing help on the keyboard commands used by the this application. Use this command to get access to the keyboard mapping's supplied with this application. If you require more general help remember to also check out the main help index page.

Help Index Command (Help menu)

The Help Index command provides quick access the main index panel of the help system. From the main help index it is easy to find information on almost all aspect of the operation of this software.

Quick Configuration Command (Help menu)

The Quick Configuration command displays the Quick Configuration dialog. This dialog provides allows you to specifying the third party help files that are to be incorporated in the Quick Search features of this application.

Quick Help Command (Help menu)

The Quick Help command provides quick help searching on the current word or currently highlighted text. It does this by searching for the particular text in the keywords of all the installed third party help files. To use this search facility place the cursor on any word or highlight a particular portion of text and use the Help | Quick Help menu option to search for the keyword. The quick help feature can also be activated by the quick help toolbar button. If no text is highlighted the quick help by default will search for help on the word currently under the cursor. In all cases the quick help searching algorithm ignores case sensitivity when searching for matching keywords.

Quick Search Command (Help menu)

The Quick Search command presents the user with the Quick Search dialog. This dialog contains the results of the last run Quick Search or can be used to enter new search details.

The quick help engine allows you to search any WinHelp file for a specific key word, provided it has been previously configured for use. In all cases the quick search algorithm ignores case sensitivity when searching for matching keywords. For more information select one of the items listed below.

Quick Help Dialog

Quick Search Dialog

Quick Configuration Dialog

About Zeus Command (Help menu)

The About Zeus command provides general information regarding the Zeus application.

Quick Help Dialog

This dialog is used to display the results of the quick help searching for the given text. This dialog is shown if the quick help search found the text in more than one of the install help files. To open any of the help file just double click on the given item or select the item from the list and hit the OK push button. For information on installing help files into the system refer to the Quick Help Configuration section of this help file.

Quick Search Dialog

This dialog is used to perform a quick help search for a particular keyword. To use this dialog just enter a keyword and then hit the OK button. The results of the search will be displayed in the results listbox. To open any of the displayed help files just double click on the item or select the item from the list and hit the OK push button. For information on installing help files into the system refer to the Quick Help Configuration section of this help file.

Quick Configuration Dialog

This dialog is used install WinHelp files so that they will be searching using the quick help search feature of this software. To install a help file first locate the WinHelp help file required then select the add button to complete the task. Help files can also be de-installed by first selecting the required file from the list of installed help files and then hitting the remove button. The quick help will search the list of installed help files looking for a match to the text supplied. If the text is found in more than one of the help files the Quick Help results dialog is displayed listing all the help files that contain the text as a keyword. You can add in as many help files as you wish but obviously the more files add the longer the time required to search through the list.

Important Note: The Remove button only removes the help file from the Zeus quick help search engine and in all cases the original WinHelp file remains untouched. Also for the search to work correctly the installed WinHelp help file must remain in its original location. If for some reason the help file is ever moved or delete the Quick Help Configuration will need to be re-run to re-install for the help file in question.

About Zeus for Windows

Zeus for Windows is a commercial shareware software product. Being shareware you can try out the software before you decide to pay for a registered version. Trying out the software allows you to determine whether it suits your needs and whether it runs correctly in your specific environment. The shareware version of Zeus and the registered version of Zeus are almost identical in functionality apart form a few minor limitations. The main difference is that the registered version is free of all the registration reminder message dialogs. At the end of the trial period if you find that the software suits your needs and you decide to continue using it you should purchase a licensed copy of the software. If you decide not to register the software it should be deleted from your system. Zeus has been specifically designed for the programmer working in the Windows 3.x, Windows 95 or Windows NT environments. This editor supports features including, a fully configurable keyboard, unlimited undo/redo, configurable color syntax highlighting, quick help searching for key words through multiple WinHelp help files, MDI multiple document interface, cut, copy and paste to clipboard, smart syntax indenting, template keyboard expansion, File Manager associations, File Manager drag and drop, background compiler and project support, in line error correction, column, block and stream marking modes for both keyboard and mouse and much more.

If you have any problems or suggestions regarding this product please let me know as I am always interested in any feedback, be it good or bad. I am also looking for suggestions as to what features should be added to future versions of Zeus for Windows.

You will also find lots more information on the Zeus home page:

http://ourworld.compuserve.com/homepages/jussi/

Also if you require a general purpose text editor take a look at the Zeus Lite for Windows text editor and once again check the home page above for more details.

Thank you for taking the time to evaluate Zeus for Windows programmers editor. For those user who decide to register thank you for supporting the on going development of Zeus for Windows.

What Is Shareware

Shareware is commercial software that you can try out before having to pay for it. Shareware is distributed by Internet shareware sites, BBS's (bulletin board systems), on diskette or CD-ROM by distributors, or by copies passed around among friends.

Trying out the software allows you to determine whether it suits your needs and whether it runs correctly in your specific environment. At the end of the trial period, if the software is suitable and if you decide to continue using it, you should purchase a licence. The shareware version of software and its documentation is complete. No essential piece of functionality has been removed or greatly modified. This should help you to make your registration decision, as you will have full knowledge of the all the facts.

However, the registration not only gets you the latest version of the product, sometimes it gives you newly introduced functions and/or bonus software. It also allows you to get support and/or provision of new versions and/or a printed manual.

The benefit of the shareware system is that it stimulates the creation of software, giving you the user a wide choice of software and also an opportunities to try out the program. It also allows software to be offered at an affordable prices and in most case you will get better software support, due to a direct relation between the author and the user. A number of products that are now available on the market would never have been born without this type of distribution.

By registering these products you are helping to promoting the creation of affordable-priced software, and, in this way, you are helping yourself.

Limitations of Shareware Version

When using the unregistered version of the software you may find a few limitations in the shareware version of the software. These are specific to the unregistered version of the software only and ***no such limitations*** apply to the registered version of the software. Here is a list of these limitations:

1. The unregistered reminder messages box is always shown at the time of start up.
2. On start up a background CRC check is run against the ZEUS.EXE file and as such the unregistered version of ZEUS.EXE can not be modified.
3. The software will prompted you with the occasional unregistered reminder messages box.
4. The File menu history has a limit of 2 files as apposed to normally being set to 9.
5. The file restart on start up feature is limited to 3 files as apposed to normally being set to 9.
6. The quick help search facility is limited to accessing the first item found only.
7. The electric keyword expansion will only allow a maximum of 8 electric keywords to be added.
8. The compiler, tools, macros and template options are limited in the number of items that can be defined.

Zeus Web Page

The Zeus web page contains information about the most recent version of Zeus, frequently asked question about the Zeus family of editors and information about the many other software products on offer. If your are looking for Shareware you will also find links to all the major shareware sites.

For anyone involved in software development for the Win16 or Win32 platforms you may be interested in the links to several software development web sites or the easy access to the Microsoft Knowledge Bases search engine.

Finally, if you are looking for some sample code you will find lots of ***FREE*** programs including a C like macro interpreter, a Windows 3.x check for DLL's utility and a host of other tools and utilities. Best of all they and all come with full source code and the executable.

To visit the Zeus home page just point your browser to:

http://ourworld.compuserve.com/homepages/jussi/

Installation Procedure

If you received Zeus for Windows on diskette (registered):

1. Insert the Zeus for Windows diskette #1 into A: drive and run the A:\SETUP.EXE installation program.
2. Choose the directory in which to install the software and press the OK button. Zeus for Windows will be installed in the directory specified.

If you received Zeus for Windows in a zip file (unregistered):

1. Unzip the contents of the zip file into a temporary directory and run the SETUP.EXE installation program from the temporary directory or the newly created diskette.
2. Choose the directory in which to install the software and press the OK button. Zeus for Windows will be installed in the directory specified.

To remove Zeus from your system either run the uninstall software located in the Zeus for Windows program group or alternatively you can also remove Zeus from your system using the manual deinstallation procedure .

Deinstallation Procedure

If you do not wish to use the automatic Zeus uninstall feature it is also possible to remove Zeus manually. As Zeus does not change any system INI file or install any DLL's in the system directories the de-installation procedure is very simple. To remove the software perform the following steps:

1. Delete the Zeus for Windows program folder using Program Manager or the desktop.
2. Delete the Zeus installation directory using File Manager, Explorer or a command line prompt.
3. Delete the 'ZeusScript' registry entry using the REGEDIT.EXE utility.
4. Delete the 'ZeusDocument' registry entry using the REGEDIT.EXE utility.

Support for Long File Extensions

Although the 32 bit version of Zeus supports long file names the extension profile feature does not support long file extensions and the maximum extension length is limited to 3 characters. This is due to the fact that by design the Windows 3.x, Windows 95 and NT versions of Zeus all support interchangeable data files. The long file extensions would cause problems with this design and as such they have not been supported. This feature will be enhanced in the future versions of Zeus once the '8.3' FAT file naming structure is made obsolete.

Having said this a profile can still be created for a long file extensions. For example to create a profile for the extension ".source" all that needs to be done is to defined a profile for the ".sou" extension. In other words Zeus only checks the first 3 characters when looking for a matching extension. The only limitation with this method is that Zeus would not distinguish files with similar long file extensions. For example the ".south", ".source" and ".soul" extensions would all share the same ".sou" profile.

Extension

Supply the three letter extension profile to be created.

List of Extensions

List of all the extension profiles that have been defined.

New

Used to create a new syntax highlighting extension profile.

Edit

Used to edit an existing syntax highlighting extension profile.

Clone

Create a new file extension profile by copying the details from the file extension profile selected.

Delete

Used to delete the currently selected extension profile.

Extension Options Dialog

The Extension Options dialog provides a method of managing the syntax highlighting extensions profiles by letting the user create, edit or delete existing extension profile definitions. When Zeus loads a file it checks to see if a syntax highlighting profile exists for the extension of the file being loaded. If such a profile exists Zeus will use this profile definition to display the contents of the file. If no profile exists then the default profile definition is used to display the file. For more information on each of the different sections of this dialog click on one of the items listed below.

Extension

List of Extensions

New

Clone

Edit

Delete

OK

Cancel

Help

Note that although the extension is limited to 3 characters Zeus will still perform syntax highlighting on files that have extensions longer than 3 characters.

Preprocessor

Supply a character that defines the pre-processor symbol.

Description

Supply a short description of the file extension profile.

String

Supply a character that defines the quoted string symbol.

Line Comment

Supply the one or two characters that define a line comment.

Comment Start

Supply the one or two characters that define the start of a block comment.

Comment End

Supply the one or two characters that define the end of a block comment.

Delimiters

Supply a string that contains all the delimiting characters.

Keyword

Enter a keyword that is to be added to the list of keywords.

List of Keywords

This list contains all the keywords defined for this extension profile.

Add

Add the keyword to the list of keywords.

Delete

Delete the currently selected item form the list of keywords.

Update

Update the currently selected item to the list of keywords.

Literal

Supply a character that defines the symbol to be used to denote a literal character. For instance in the C/C++ language the '\' is used as the literal character in the following examples:

'\0' or '\n'

Keyword Prefix

Supply a character that defines the symbol to be used to denote a keyword prefix. This character is used for languages (like LaTex and HTML) that have special keyword delimiting characters. For example the HTML language would use the '<' keyword prefix character. This allows the following type of code to be correctly colorized:

```
<TD><B><A HREF="index.htm">Main<BR>Page</A></B></TD>
```

For most other languages this field should be left empty.

Keyword Postfix

Supply a character that defines the symbol that is to be used to denote a keyword postfix. This character is used for languages (like LaTex and HTML) that have special keyword delimiting characters. For example the HTML language would use the '>' keyword postfix character. This allows the following type of code to be correctly colorized:

```
<TD><B><A HREF="index.htm">Main<BR>Page</A></B></TD>
```

For most other languages this field should be left empty.

Numbers

Use this field to define a set of characters that are to be interpreted as numbers for this particular file extension.

Functions

Use this field to define the regular expression to be used to locate functions for the given file extension. For example if you where defining a PASCAL file extension you could use the following regular expression:

^procedure +[_a-z0-9]+

Or if you where defining a Windows resource file extension you could use the following regular expression:

^[0-9a-z_A-Z]+DIALOG

Ignore Case

Use this check box to indicate that the keywords supplied are to be treated as case insensitive.

Extension New Dialog

The New Extensions dialog allows the user to create a new syntax highlighting extension profile definition. If the profile being created is for a language extension like LaTex or HTML them make sure the keyword prefix and postfix characters are defined correctly. For more information on each of the different sections of this dialog click on one of the items listed below.

Preprocessor

Description

String

Literal

Line Comment

Comment Start

Comment End

Delimiters

Numbers

Keyword Prefix

Keyword Postfix

Functions

Keyword

List of Keywords

Add

Delete

Update

Ignore Case

OK

Cancel

Help

Extension Edit Dialog

The Edit Extensions dialog allows the user to edit an existing syntax highlighting extension profile definition. If profile being edited is for a language extension like LaTex or HTML them make sure the keyword prefix and postfix characters are defined correctly. For more information on each of the different sections of this dialog click on one of the items listed below.

Preprocessor

Description

String

Line Comment

Comment Start

Comment End

Delimiters

Keyword Prefix

Keyword Postfix

Functions

Keyword

List of Keywords

Add

Delete

Update

OK

Cancel

Help

Tools Command (Options menu)

The Tools command will display the Tool Options dialog. This dialog allows the user to add third party tools to the Zeus Tools menu . It also allows existing Tool menu items to be removed or edited.

Macros Command (Options menu)

The Macros command will display the Macro Options dialog. This dialog allows the user to add macros to the Zeus Macros menu. It also allows any existing Macro menu items to be removed or edited.

Project Command (Options menu)

The Project command will display the Project Options dialog. This dialog allows the user to configure the project handling features provided by Zeus. Once the project has been successfully configure the project can be easily managed using the Project menu commands.

Compiler Command (Options menu)

The Compiler command will display Compiler Options dialog. This dialog allows the user to configure the compiler features provided by Zeus. Once the compiler has been successfully configure compilation is easily done using the Compile menu command.

Font Command (Options menu)

The Font command will display the Font dialog. This dialog allows the user to configure the font to be used when displaying the document text.

Colors Command (Options menu)

The Colors command will display the Colors Options dialog. This dialog allows the user to configure the colors that are to be used for displaying the document text.

Editor Command (Options menu)

The Editor command will display the Editor Options dialog. This dialog allows the editing behaviour of Zeus to be easily configured.

Filters Command (Options menu)

The Filters command will display the Filters Options dialog. This dialog allows the user to configure additional file open and save as extension filters. These filters can then be used to help in the filtering process when using the open and save as dialogs.

Spelling Command (Options menu)

The Spelling command will display the Spelling Options dialog. This dialog allows the user to configure the built in spelling engine.

Keyboard Command (Options menu)

The Keyboard command will display the Keyboard Options dialog. This dialog allows the user select one of the pre-defined keyboard maps, modify an existing keyboard map or to define a new keyboard mapping.

Templates Command (Options menu)

The Templates command will display the Template Options dialog. This dialog allows the user to define keyboard templates that aid in process of entering pre-defined coding constructs.

Extensions Command (Options menu)

The Extensions command will display the Extension Options dialog. This dialog allows the user to create, modify or delete a file extension profiles. The user can define their own profile or select form one of the pre-defined C/C++, Pascal, Cobol, Clipper, ADA, etc. etc. extension profile definitions.

Capture Standard Error

This option tells allows Zeus to capture the standard error output produced by the executable. By default Zeus is setup to only capture the standard output and not the standard error information. For this option to work correctly the Zeus install directory ***must be*** in the system path (see the AUTOEXEC.BAT file).

Use Zeus Run File

This option tells Zeus to use the ZEUSCC.PIF when running the external compiler (Note: this file is used internally by Zeus and should not be changed or deleted). If this mode is selected the user must also enter the command line required to run the compiler in the entry field provided. This PIF file is copied to the Zeus directory at the time of installation.

Use Other File

This option allows Zeus to use an alternative batch file to run the external compiler. This mode is not normally used but is offered as an alternative method of controlling the compile process. Before Zeus runs the compiler it changes the current directory to match that of the file being compiled and then spawns this compiler batch file. It passes the compiler batch file three arguments via the command line. Argument #1 is the name of the file being compiled and argument #2 is the error file that needs to be generated during compilation and argument #3 is the name of the file being compiled without extension. For the compile to work correctly the batch file must redirect the compiler errors produced to the error file specified. As an example of a suitable batch file refer to the CC.BAT example located in the Zeus installation directory. To use CC.BAT check the 'Use other file'' option and enter CC.BAT as the compiler run file.

Run File

When using the Use Other File option this entry field is available for entering the command that is to be run as an alternative to the internally generated Zeus compiler run file.

Help to Debug the Session

Use this option to help debug the compile or build process. This option will put a pause in the batch file and run the process in a window. This should help with the debugging of the compile/build process as it allows the user to see what is being run and why the command is failing.

Display Mode

You can use these options to change the way Zeus manages the compiler DOS session. Normally you would run the compiler minimised or hidden but if you are having problems use the normal or maximised display modes as these modes will allow you to see what is going on during the compile process.

Command Line

Enter the typical command line needed to compile a file called SAMPLE. To check the command line entered try using the same command line from any DOS session. This enter filed is only available for the 'Use Zeus File' run mode (ZEUSCC.PIF). For the 'Use Other Compiler' case the command line entry field is disabled as the real command line will be located internally to the actual batch file supply.

DebugCommandLine

Enter the typical command line needed to run the debug tool for the executable produce by the project build cycle. To check the command line entered try using the same command line from any DOS session.

Add

This button allows you to add a new command line to the list of commands.

Delete

This button allows you to delete the currently selected command line from the list of commands.

Update

This button allows you to update the currently selected command line.

Errors Regex

Use this entry field to define a regular expression that is to be used to search for errors in the compiler/builder generated output. For example some compilers/builders add a file to each error string produced and thus the following regular expression could be used to locate these errors:

[^ \t\")(<'`]*[a-zA-Z0-9_]*\.[a-zA-Z0-9_]*[^ :\t\")(>'`*}]

Lines Regex

Use this entry field to define a regular expression that is to be used to search for the line numbers in the compiler/builder generated output. For example some compilers/builders add the word ***line*** to the line number information so the following regular expression could be used to locate the line number from the error string:

line [0-9]+

Project Options Dialog

The Project Options dialog is used to configure the Zeus project build support. Once you have configured the project options use the Project menu to control the build process. For more information on each of the different sections of this dialog click on one of the items listed below.

Use Zeus File

Use Other File

Run File

Help to debug the session

Display Mode

Command Line

Debug Command Line

Errors regex

Lines regex

Capture standard error

Update

Cancel

Help

Remember that the you can also include one of the many Zeus TAG macros in the project command line and the debug command line entry fields. If you have any problems getting the project to build correctly refer to the Understanding the Project Support section for more details. If you are running on OS/2 2.1 WinOS/2 or OS/2 3.0 Warp the project support will not operate correctly. Please refer to the known problems section for more details.

Compiler Options Dialog

The Compiler Options dialog is used to configure the Zeus compiler support. Once you have configured the document compile option use the Compiler menu to control the compile process. For more information on each of the different sections of this dialog click on one of the items listed below.

- Use Zeus File
- Use Other File
- Run File
- Help to debug the session
- Display Mode
- Command Line
- Errors regex
- Lines regex
- Capture standard error
- Add
- Delete
- Update
- Cancel
- Help

Remember that the you can also include one of the many Zeus TAG macros in the compiler command line entry field. If you have any problems getting the compiler support to work refer to the Understanding the Compiler Support section for more details. If you are running on OS/2 2.1 WinOS/2 or OS/2 3.0 Warp the compiler support will not operate correctly. Please refer to the known problems section for more details.

Font

Type or select the name of the font that is to be used for displaying the document text. Zeus will only list fonts that are mono-spaced. Note that proportional fonts are not supported.

Font Style

Select the font style that is to be used for displaying the document text.

Font Size

Type or select the point size of the font that is to be used for displaying the document text.

Font Dialog

The Font dialog is used to select the font that is to be used for the display of the document text. For more information on each of the different sections of this dialog click on one of the items listed below.

Font

Font Style

Font Size

OK

Cancel

Important Note: The font used for printing will be a font that is the nearest match to the currently selected display font and will never be the exact same as the font as the font used to display the document text. What this all means is that the print out is not WYSIWYG (what you see is what you get). All that is guaranteed is that the print out will be easy to read.

Display Mode

This setting determines if Zeus will create a new document as tiled, minimised, maximised or if the document is created using the default MDI behaviour.

Tabs as spaces and Tab Size

These options let you set the tab size and determines if tabs should be inserted or if spaces characters should be used instead of tabs. Please note that Zeus does not support tabs stops at irregular intervals and when tabs are inserted as tab characters the cursor movement will not split a tab character.

Line Wrap

With line wrap enabled Zeus will automatically insert a carriage return line feed when the text entered exceeds the column position specified.

Automatic backup

This option enables or disables the background backup processing. The background time interval can be modified by entering a suitable number in the backup edit field. The number represents the backup interval in minutes. If a value of zero is entered no backup is performed. The background backup is only activated for files that have been changed and that where also last backed up more than the backup interval minutes earlier. If both these conditions are met then the file is copied to the zBackup directory (located relative to the Zeus installation directory). This feature is meant as a safe guard against the possible loss of data in the unlikely event that the Zeus editor should crash. Should you find that the editor did crash then there is a chance that a backup file may been created and as such there is a chance that you will only loose a small amount of work, which is better than nothing :).

LoadBackup

This option enables or disables the backing up of files as they are loaded for editing. If this option is enabled then every time a writable file (ie not read only) is opened for editing a copy of the original file will be made to the backup directory.

Search Path

The search path is the path that is used when Zeus tries to load a file automatically. Enter in the path that is to be searched for automatic file loading. For example:

```
c:\cpp\include;c:\cpp\owl\include;d:\msvc\drc\include;
```

Also search INCLUDE variable

This option forces Zeus to search the INCLUDE environment variable when searching for files that are to be loaded automatically.

Smart cursor indenting

This option enables or disables the smart indenting. When enabled Zeus will automatically perform cursor alignment based on the information contained in the lines above. This option is designed to minimise the amount of cursor positioning required when entering code or text information.

Smart cursor braces

This option enables or disables the smart cursor braces. If enabled Zeus will add a matching brace and position the cursor at an indented position. This action is activated when the enter key is pressed and the last character of the line is a brace character. The positioning of the braces is controlled by the 'Normal brace' and 'Indent brace' options.

Smart templating

This option enables or disables the smart Zeus template feature. If enabled Zeus will attempt to perform smart templating based on the templates that have been defined using the Options | Template menu pull down. If enabled the smart templating is always triggered by the space key.

Indent braces

This option is used in conjunction with the Smart cursor brace option. When selected the smart cursor brace feature will produce indented brace output similar to the examples shown below.

```
for (int i; i < 0; ++i)
        {

        }

for (int i; i < 0; ++i)       {

        }
```

Normal braces

This option is used in conjunction with the Smart cursor brace option. When selected the smart cursor brace feature will produce normal brace output similar to the examples shown below.

```
for (int i; i < 0; ++i)
{

}

for (int i; i < 0; ++i) {

}
```

Restore files on startup

This option enables or disables the restoration of files on startup. If enabled Zeus will restore the files that where open as when Zeus was last used.

Restore size on startup

This option enables or disables the restoration of size on startup. If enabled Zeus will restore to the same size as when Zeus was last used.

Restore last working directory

This option enables or disables the restoration of the work directory on startup. If enabled Zeus will restore to the same directory as when Zeus was last used.

Display a left margin

This option turns on or off the left hand margin. When enabled Zeus adds a fixed size left hand margin to the display. The size of the margin is fixed to 4 pixels.

Brief column paste mode

This option enables or disables the Brief emulation column paste feature. If enabled Zeus perform column paste cursors movements similar to the original Brief editor. This option is provide to allow a choice in the type of column paste cursor movement required.

Display Alt key codes

This option enables or disables the display of Alt key codes. If enabled Zeus will paint any Alt key that is typed (and not bound to a function) otherwise Zeus will ignore these keys.

Display Ctrl key codes

This option enables or disables the display of Ctrl key codes. If enabled Zeus will paint any Ctrl key that is typed (and not bound to a function) otherwise Zeus will ignore these keys.

Read file as Unix

This option enables or disables the UNIX file read mode. If enabled Zeus will read the file as UNIX files (as apposed to the MS-DOS format).

Write file as Unix

This option enables or disables the UNIX file write mode. If enabled Zeus will write the file as UNIX files (as apposed to the MS-DOS format).

Lines insert after cursor

This option controls the way Zeus handles line inserts when pasting lines of text from the clipboard. If enabled Zeus perform line inserts after the current cursor position otherwise Zeus will insert the lines before the current cursor position. This option is provide to allow a choice in the type of line pasting that is required.

Allow multiple file copies

This option allows Zeus to open multiple versions of the same file. If enabled Zeus can maintain multiple copies of the same file with each version being independent of the others. But note that for such multiple copies the last saved instance of the file will be the version that is left on the disk. When disabled Zeus will only ever allow one copy of the file to be open at any one time.

Read File As OEM

This option forces Zeus to perform OEM text translation on the file as it is loaded for editing.

Write File As OEM

This option forces Zeus to perform OEM text translation on the file as it is written back to the disk.

File to open must exist

This option forces the file entered in the File Open dialog to exist. If this option is set and a file name entered refers to a file that does not exist an error message is displayed. In the case where this option was disabled the non existent file would be loaded for editing but an empty document would be created.

Editor Options Dialog

The Editor Options dialog allows you to change the basic behaviour of the editor. There are also a few extra INI file settings that can be used to configure the editor. For more information on each of the different sections of this dialog click on one of the items listed below.

Display Mode

Tabs as spaces and Tab Size

Line Wrap

Automatic backup

Make backup when file is loaded

Search Path

Also search INCLUDE variable

Smart cursor indenting

Smart cursor braces

Smart templates

Indent braces

Normal braces

Restore files on startup

Restore size on startup

Restore directory on startup

Display a left margin

Brief column paste mode

Allow multiple file copies

Lines insert after cursor

Display Alt key codes

Display Ctrl key codes

Read file as Unix

Write file as Unix

Read file as OEM

Write file as OEM

File to open must exist

Allow uppercase words

When enabled, this option will convert uppercase words to lower case before the word is spell checked. If this option is disabled the word will be checked in its capitalised form but for such a word to be correct the same word with the same capitalisation must also exist in the dictionary.

Allow words containing numbers

When enabled, this option assume words that contain numbers are correct and as such will not need to be checked for spelling.

Allow e-mail address details

When enabled, this option assumes that HTML and internet address strings are correctly and as such will not need to be checked for spelling.

Words of one character length assumed correct

When enabled, this option assumes that words of one character length are correct and as such will not need to be checked for spelling.

Use built-in English grammar rules

When enabled, this option uses the built in grammar rules coded into the spell checker. Use this option with great care as it applies some very generic rules to the words when checking for spelling (in the form of word prefixing). There are undoubtedly several problems with these built in grammar rules since in general they just apply very simple rules, yet unfortunately the English language has never been quite as simple.

Set As Default

This button sets the currently select dictionary to be the default dictionary. The default dictionary is the dictionary that is used by the spelling engine as its source of words. The dictionaries list represent all the currently installed dictionaries as located in the zSpelling directory.

Default Dictionary

This field indicates which is the current default dictionary. The default dictionary is the dictionary that is used by the spelling engine as its source of words.

Spelling Options Dialog

The Spelling Options dialog is used to configure the Zeus spelling support. At present only an American/English dictionary is available. Other dictionaries may be produced depending on demand. Once the spelling has been correctly configured you can use the Spelling menu commands to check for correct spelling. For more information on each of the different sections of this dialog click on one of the items listed below.

Allow uppercase words

Allow words containing numbers

Allow e-mail address details

Words of one character length assumed correct

Use built-in English grammar rules

Set as default

Default dictionary

OK

Cancel

Help

Find Dialog

This dialog lets you locate a particular string in the currently active document. A history list of the most recently entered search text is also provided. The search can be made case sensitive, word sensitive or can use regular expressions by just selecting the appropriate check box. The direction of the search can be specified using the radio button options. The search text entry field also accepts special control codes to help control the effect of the search. For more details refer to the Advanced Search and Replace Features section. For more information on each of the different sections of this dialog click on one of the items listed below:

Match whole word

Match case

Regular expression

Search up

Search down

Find

Cancel

Help

Replace Dialog

This dialog lets you find and replace an existing string with an alternative string. A history list of the most recently entered search and replace text is also provided. The replace action can be made case sensitive, word sensitive or can use regular expressions by just selecting the appropriate check box. The direction of the replace can be specified using the radio button options. The search and replace text entry fields also accept special control codes to help control the effect of the search. For more details refer to the Advanced Search and Replace Features section. For more information on each of the different sections of this dialog click on one of the items listed below:

Match whole word

Match case

Regular expression

Search up

Search down

Find

Replace

Cancel

Help

Match Whole Word

The search will only match whole words and not partial words.

Match Case

The search will only match words with the same case.

Regular Expression

The search will use regular expression rules.

Replace All

Replace all occurrences of the find text with the replace text based on the current directional setting.

Search Up

Search in an upward direction from the current cursor position.

Search Down

Search in an downward direction from the current cursor position.

Find Text

Search for the find text entered.

Replace Text

Replace the "find text" found with the replacement text.

Regular Expressions

Zeus for windows supports pattern matching using the regular expression notation similar to that found in the UNIX environment. This is similar but not identical to the regular expression notation used by the Brief editor. Below you will find a short tutorial describing what regular expression are all about. Also remember check out the many examples of actual regular expressions also provided.

Regular expressions provide a notation that allows the user to better describe the text that is to be searched for thus enabling a precise match of the text to be carried out. At first glance regular expressions can appear difficult to understand and hard to use but once you have a basic understanding of how to construct these regular expressions you will find them a powerful extension to your editing arsenal.

In the section that follows, you will be given a description on how to use regular expressions. To get the best use out of this material it is best that you try the example searches provided. Run the search against the following test string:

Example Line:

```
This is a test. A line with a p and ppp groups
```

Also note that the search can be both case sensitive or case insensitive depending on the current case sensitivity setting. The examples given below are assuming that the all the searching is done without case sensitivity.

Regular expressions are formed by the concatenation of several rules. The regular expression engine will then run these rules in an effort to locate any text that matches the rules supplied. For example to search for the word 'orange' you would use the rule 'orange', which is exactly the same as if you had used any standard search method. So the first thing to remember is that by default regular expressions behave just like the standard search method. The power of regular expressions is derived from the use of its special operators, which are summaries below.

The **'*'** operator matches zero or more occurrences of that preceding expression. For example to find a string containing any number of p's you would use the following expression:

p*

The **'+'** operator matches one or more occurrences of that preceding expression. For example to find a string containing one or more p's you would use the following expression:

p+

The **'?'** operator matches zero or one occurrences of that preceding expression. This is the same as saying the expression is optional. For example to find a string containing one or no number of p's you would use the following expression:

p?

Now if you run these examples on the test search string you will get some very different results.

The **'.'** operator matches any single character. For example to match any word starting with 'g' that is five characters long but ending in 's' you would use the following expression:

g....s

This will locate the word 'groups' in our test search string.

The **'|'** operator matches one expression or the other. For example to find all line or groups we would use the following example:

line|groups

The **'(',')'** operators provide a method for grouping regular expressions. As an example consider the case of trying to match a string that can have at least one 'i' or 'p' and a following 's' we could write the following expression.

(i[a-zA-Z]*s)|(p[a-zA-Z]*s)

The **'(',')'** characters also provide some control over the order of execution. For example to find any characters that are between 'a' and 'e' or 'm' and 'q' but located at the end of the line the expression would be:

(([a-e]+)|([m-q]+))$

Finally the **'(',')'** operators play an important role in the advanced search and replace features of this editor.

The **'^'** operator means that the pattern must match the start of the line. For example the following search will only find the 'test' string that is at the start of the line and not 'test' strings in the middle of the line.

^test

The **'$'** operator means that the pattern must match the end of the line. For example the following example will only find the letter 's' that is at the end of the line.

s$

The **'[',']'** operators can be used to define a set or group of characters that must be matched. For example to find any of the letters 'a','e','i','o' and 'u' you would write the following regular expression.

[aeiou]

To help define groups the **'-'** Operator can be used within a group definition. For example to find all the letters between 'a' an 'l' you could use the following ranged group expression:

[a-l]

The **'^'** operator takes on special meaning if used as the first character of a group definition. In this case it no longer signifies a start of line, but is used to represent a negated grouping. For example to find all characters that are not vowels we would use the following (ie find all characters that are not in this group).

[^aeiou]

The **'\'** operator allows you to define one of these special characters as a literal character. For example if you need to find the '.' character in the search text you would use the following expression:

\.

Without the '\' delimiting character, if you had just done a search on the '.' character you would have found every character in the line.

NOTE: The **'\'** operator allows you to search for tab characters using the **'\t'** search pattern in both regular and normal search modes Also note that Zeus ***does not*** support the concept of using **'\n'** to search for the carriage return line feed character.

The power of regular expressions comes when the expressions are concatenated to form a more complex expression. For example Zeus uses the following regular expression to locate a C function definition but not a function prototype.

^[_a-z0-9]+[&*\t]+[_a-z0-9 \t]*[_a-z0-9]+[\t]*[(]+.*[^;]+$

Using this expression the following line is ignored as a function prototype:

long PASCAL WndProc(HWND, WORD, WORD, LONG);

But the following line is identified as a valid function definition:

int PASCAL WinMain(HANDLE hInst, HANDLE hPrv, LPSTR psz, int nCmd)

At first glance this expression seems far to complicated, but it can be better understood by just looking at the expression as a collection of collection of smaller, simpler expressions:

Return White cdecl Function White Arguments

^[_a-z0-9]+[&*\t]+[_a-z0-9 \t]*[_a-z0-9]+[\t]*[(]+.*[^;]+$

Look at this expression as a collection smaller regular expressions where 'Return' is the return code, 'White' refers to any white space, 'cdecl' refers to things like the PASCAL as shown in the example, 'Function' is the function name and 'Argument' is the argument list provided the line does not end in a ';' character.

You can also use the regular sub-expression for search and replace operations. For more information on this feature refer to the advanced search and replace section of this document.

Advanced Search and Replace Features.

Apart from the standard search and replace features Zeus also offers the **'\'** special character to help with searching. The **'\'** character can be used to search for tab characters using the **'\t'** search string. If you need to search for the **'\'** itself then use the **'\\'** search string. When this special character is used as a part of the replacement text it takes on extra meaning (except if it is entered as **'\\'** or **'\t'** in which case it behaves as expected). In this case the **'\'** is taken to represent the actual text that was found. This characteristic can be used for example to add the string 'ing' to the word 'search'. To do so you would use the following search and replace information.

Search For: search
Replace With: \ing
Example Line: search
Result Is: searching

This feature is also available for use with regular expression searches. For example to add a the matching '"' character to the end of the 'FileReload4' you could use the following search and replace information.

Search For: [ ]+},
Replace With : **"**\
Example Line : **"**FileReload4 },
Result Is: "FileReload4" },

For the case of the regular expression search and replace operations the **'\'** operator takes on an even more powerful meaning when it is combined with the **'(',')'** operators. In this case it can also be used to indicate sub-expressions and allows for far greater control over the replace procedure.

For example consider a simple case where the following regular expression is used in a search:

(psz)(The)(String)

When a string matching this expression is found, the results of this search can be thought of as consisting of four actual parts. The first of these parts is the entire string found and for replacement purpose this can be represented by the **'\'** character. The other three parts to the search correspond to the three sub-expressions identified in the search string. These can be represented using the **'\1', '\2'** and **'\3'** characters respectively. In general and sub-expression can be expressed using the characters **'\n'** where n is a digit 1 through 9 and the value of n is determined by counting occurrences of the **'('** character starting from the left.

Note that the **'\'** and **'('** characters are no different to any of the other regular expression operators in that if the they are preceded by the **'\'** character then the lose their special meanings and will revert to the just their character literal equivalents.

As a complete example consider searching for the **(psz)(The)(String)** expression in a file that contains the following text:

pszTheString

Here are some examples of replace operations that use this sub-expressions feature:

Replace: \Test **Result Is:** pszTheStringTest
Replace: \3\2\1Test **Result Is:** StringThepszTest
Replace: \3\2\1\ **Result Is:** StringThepszpszTheString
Replace: \\\Test **Result Is:** \pszTheStringTest

Note that these sub-expressions can also be nested. For example consider searching for the **((this) is) nesting** expression in a file that contains the following text:

this is nesting

Here are some examples of replace operations that use this nested sub-expressions feature:

Replace: \ test	**Result Is:** this is nesting test
Replace: \1 test	**Result Is:** this is test
Replace: \2 test	**Result Is:** this test

Regular Expression Examples

Below are some examples of how to use regular expressions. You will notice that only small changes in the regular expression can have a big effect on the outcome of the search. If you need more help with these regular expressions you should also take a look at the introductory section on understanding regular expressions. Try the following search examples using the example line provided and observe the text found paying particular attention to the highlighted text of the search result.

Using this Example Text:

```
This is a test , tests , testing and test()
```

Search For: test[^a-z]

Description:

Find any 'test' that is not followed by a character that is not in the set 'a' through 'z' character.

Search For: test[^a-z]*

Description:

Find any 'test' followed by any number of characters that are not an 'a' through 'z' character.

Search For: test[^a-z]+

Description:

Find any 'test' followed by at least one character provided it is not an 'a' through 'z' character.

Search For: test[^a-z]?

Description:

Find any 'test' followed by at least one or none (optional) character which is not an the 'a' through 'z' character.

Now Using this Example Text:

```
This line contains some 123, .45, and 123.45 numbers as well as some
"quoted text".
```

Search For: ([0-9]+\.[0-9]*)|([0-9]*\.[0-9]+)|([0-9]+)

Description: This regular expression will find any unsigned number.

Search For: [a-zA-Z]+

Description:

This regular expression will find all words that only contain alphabetic characters.

Search For: "[^"]+"

Description:

This regular expression will find any quoted string.

Now Using this Example Text:

```
#include "zeus.h"
#include "zfwprt01.h"
#include "zfwprt01.hpp"     //-- file used to make zeus portable
```

Search For: [^e]*ab

Description:

Find any text ending in 'ab', provided it does not contain an 'e' character.

Search For: [^e]*ab$

Description:

Find any text ending in 'ab' and also at the end of the line, provided it does not contain an 'e' character.

Search For: (([1]+.hpp)+)|(([1]+.h)+)

Description:

This search will find any '1.h' or '1.hpp' file extensions.

Preset Colors

Lists all the pre-defined colors schemes.

Syntax Groups

List the different syntax highlighting groups that can be configure.

Colors

List the different colors that are available.

Color Options Dialog

The Color Options dialog allows the user to customise the colours used for syntax highlighting. This dialog allows the selection of a pre-defined colour scheme, the modification of one of the existing colour schemes or the creation of a new color scheme. For more information on each of the different sections of this dialog click on one of the items listed below.

Preset Colors

Syntax Groups

Colors

Update

Cancel

Help

Use Default

Zeus provides a default template extension that is used by all files. This option determines whether these default templates should also be used (inherited) in the definition of the new templates extension.

Description

This is the descriptive name of the template extension. For example you may be defining a template extension for HTML files.

Add

The add button adds the template expansion to the list of currently defined expansions.

Delete

The delete button removes the selected template expansion from the list of currently defined expansions.

Update

The update button updates the currently selected template expansion.

Edit Template Dialog

The Edit Template dialog allows the user to edit an existing template extension. The process involves defining a suitable template keyword and the corresponding expansion that is to be associated with this keyword. As an example of a typical expansion take a look at the template expansion example contained in this document. For more information on each of the different sections of this dialog click on one of the items listed below.

Use the default templates

Description

Add

Delete

Update

Template

Expansion

Important Note: All template keyword expanding is disabled if the editor is in overwrite mode. Also note that the \n control code will override any \b and \t controls codes when the Edit Options *'Smart cursor indenting'* feature is enabled. To overcome this anomaly always use the \t and \b control codes after the \n control code. As a final note the '*Smart cursor bracing"* feature is disabled during the actual template expansion process so the required brace indenting will need to be specified by the actual expansion.

New Template Dialog

The New Template dialog allows the user to create a new template extension. The process involves defining a suitable template keyword and the corresponding expansion that is to be associated with this keyword. As an example of a typical expansion take a look at the template expansion example contained in this document. For more information on each of the different sections of this dialog click on one of the items listed below.

Use the default templates

Description

Add

Delete

Update

Template

Expansion

Important Note: All template keyword expanding is disabled if the editor is in overwrite mode. Also note that the \n control code will override any \b and \t controls codes when the Edit Options *'Smart cursor indenting'* feature is enabled. To overcome this anomaly always use the \t and \b control codes after the \n control code. As a final note the '*Smart cursor bracing"* feature is disabled during the actual template expansion process so the required brace indenting will need to be specified by the actual expansion.

Template Field

This is keyword for the expansion. Best results are achieved if the keyword is short, unique and preferably a non English word. As an example the keyword 'for' is not a good choice, as it can easily be mistaken with the English word 'for'. A better choice would be 'fr' or 'for_' or some other form of abbreviation for of text.

Expansion Field

This is the actual string that is expanded out as a replacement for the keyword. Note that the original keyword is replaced by the expanded text. The expansion text string can contain special characters that control the way the text is expanded. The following control codes are support.

\c	Place cursor here once expansion is complete
\n	Insert carriage return line feed
\t	Insert tab character
\b	Insert back tab character

If you want to use any these special codes in your expansion but without the required effect (ie you actually want a \b in the expansion) then just add an extra '\' character to these codes. For example to get a \b in the expanded text output just add \\b to the expansion.

Template

The name of the template extension to be created or edited.

List of Templates

The list of the currently available template extensions.

New

Creates a new template extension.

Edit

Edits the currently selected template extension.

Delete

Deletes the currently selected template extension.

Template Options Dialog

The Template Options dialog allows the user to add, modify or delete keyboard templates. By definition a keyboard template is a keyword string that will be replaced by the expanded text when activated by a specific trigger. In the case of Zeus template expansion are always triggered by the space bar and ***this is not*** configurable. Zeus allows you to define templates that are specific to a particular file extension or by adding the template to the default extension the template is then made available to all file extension. For more information on each of the different sections of this dialog click on one of the items listed below.

Template

List of templates

New

Edit

Delete

OK

Cancel

Help

Creating Template Expansion

As an example of a keyword template edit the default template expansion and define the following keyword by filling in the appropriate fields of the dialog box.

Keyword: ifg Expansion: if (\c >)\n{\n\t\n};\n

To use the newly created expansion hit the Add then the OK buttons and return to any active text document window. Next enter the keyword 'ifg' (ie short for 'if greater than') followed by the space character. The space character will trigger a keyword search and since this keyword is now defined the result would be that the word 'ifg' is expanded out to read as follows (provided you are not in overwrite mode):

```
if ( > )
{
};
```

Search Criteria Used for Locating Keywords

The search engine first checks to see if the characters of the user text entered matches any of the keywords defined. The second criteria for the search is that if the text passes the first test then the text entered must also be a unique word. As an example of this consider the following case:

'for ifg<space>' will expand to:

```
for if ( >)
{
};
```

but

'forifg<space>' 'ifg<space>for'

will not expand because the search fail to find a word match.

Toolbar Information

The toolbar provides quick access to some of the more common editing functions. All the functions listed on the toolbar are also available using the keyboard or menu interface. To determine what each toolbar item actually does just place the cursor over the item in question and a description of the item will be displayed in the status bar message area.

Status Bar Information

The status bar is used to provide information about the current state of the machine and also display messages to the user. The message display region is used to display error messages, general information and help information. The current line and column position are also indicated along with the current state of the insert/overwrite mode, the CAPS Lock, NUM Lock and Scroll Lock keys.

The status bar also provides a quick source of help for the menu items and toolbar functions available. To find help on a toolbar or menu item just place the cursor over the toolbar button or select the menu item in question and then read the descriptive text displayed in the status bar message area.

Run

Use the run button to execute the macro script file entered.

Find

Use the find button to help locate the macro script file to be run.

Script

Use the script entry field to enter the name of the script to be run. If you do supply a fully qualified script then by default Zeus will look for the script file in the zScript directory (located relative to the installation directory). Also if no extension is supplied then by default the ***zm*** extension is assumed. As an example consider the case of the following script file:

c:\zeus\zscript\example.zm

Since this file is located in the default script directory and has an extension of ***zm*** then all that is needed to run this script would be to supply the script file name which in this case is ***example***. All the other details will be taken care of by the Zeus.

Execute Macro Script Dialog

The Execute Macro Script dialog allows you to run a Zeus macro script file. For more help on the writing macro scripts refer to the macro script section of this online help. For more information on each of the different sections of this dialog click on one of the items listed below:

Script

Run

Find

Cancel

Remember that the you can also include one or more of the many Zeus TAG macros as command line arguments to the macro script file.

Writing Macro Scripts

It is possible to configure Zeus by writing specialised macro scripts. For the macro scripting to work it is assumed that the Zeus install directory has been added to the system path. If you have any problems running the ZMI.EXE ***please check*** that this is in fact the case. For more information on what the Zeus macro scripting language offers see the information provided below.

Bugs, Contributions or Suggestions

The macro scripting has only recently been added to the Zeus editor and as such it is effectively still version 1.0 code. Because the code is relatively new there will undoubtedly still be several bugs waiting to be found. If you managed to find a problem please don't hesitate to send in a bug report. The easiest way of reporting a macro bug would be to provide the text for the offending macro along with a short description of the actual problem.

If you write a macro that you would like to share with others I would be more than happy to make your macro available on the Zeus web page:

http://ourworld.compuserve.com/homepages/jussi/zMain.htm

To submit a macro all you need to do is e-mail the macro that you would like to publish. Please make sure that as a minimum the macro has a full header description similar to the following example.

```
//-- Name: Example
//-- Description: This is a very simple macro that does
//-- nothing. If needed, I could have got carried away
//-- with a long description.
//-- Author: Jussi Jumppanen
//-- Contact: xidicone@iname.com or
//--  http://ourworld.compuserve.com/homepages/jussi
int Example()
{
   //-- don't forget to comment your macro
   return 1;
}
```

Finally if you have and suggestions on how to improve the macro support please let me know. This may include suggestions on new functions or new features. There are already plans to have support for macro function calls (i.e. macros that can call other macros) and to also improved the macro error handling and runtime error reporting.

What are Macro Scripts

The Zeus macro script language is a C like language that not only allows you to easily extend the functionality of the Zeus editor but also gives you a way of automating repetitive tasks.

A macro script is a collection of commands and logical flow instructions that are grouped together as a single command in the form of a script file. If you perform a task repeatedly, you can automate the task by writing a macro script. Thus instead of manually performing a series of commands it is possible to automate the task by just running the macro script instead.

Zeus offers a standalone macro interpreter called ZMI.EXE as well as an interpreter that is built into the editor itself. For an example of a macro script run using the standalone interpreter see the Macro Interpreter Example located on the Tools menu. For information on the built in interpreter refer to the Execute Macro Script dialog.

Macro Language Syntax

The Zeus macro language uses a small subset of the C language. The language supports several of the C flow constructs including the ***for***, ***if*** and ***while*** instructions. The variable types supported are the ***char***, ***int*** and a new type called a ***string*** variable type. The ***string*** type is very similar to the C language char array or a char * types and can be treated as a such. The string type has been designed to take care of the process of allocating and freeing the memory needed to hold the string variables which is essential since the Zeus scripting language does not support the concept of pointers.

In general variables do not need to be declared but by default any undeclared variables are automatically declared as type ***int*** variables. Obviously this means that if a you require a ***string*** or ***char*** variable type you will need to explicitly declare the variable. Note that the scope of all variables is global and the concept of local variables with a local scope is not supported.

One of the best methods of learning how to use the Zeus macro language would be by studying the many examples provided.

Builtin Macro Functions

The Zeus macro interpreter is available in two forms. Firstly there is the Zeus Editor interpreter, that is built into the editor itself and then there is the ZMI.EXE interpreter, which can be run from any DOS command line prompt (or as a tool from the Zeus Tools menu). Although both interpreters use the same language syntax, in general their macros are not compatible due to the fact that they support different sets of built in functions.

For this reason the interpreter functions fall in to three broad categories. The first group of functions are made up of the Standard C Functions and these are common to both the interpreters. A second group of functions are specific to the Zeus Editor interpreter and include all the keyboard functions and document manipulation functions. Finally there is a group of DOS related functions and these can only be used by the ZMI.EXE DOS command line interpreter. set of functions standard C like functions that are common to both interpreters.

Standard C Like Macro Functions

These are a set of standard C like functions that are common to both interpreters. In general all functions will return 1 on success and -1 if they encounter an error condition. The syntax for the functions is very similar to the syntax of the standard C library functions on which they are modelled. Below is a list of the functions provided:

int access(string szFileName, int sCode);

Get the access details for the given file. The code can be

- 6 Check for read and write permission
- 4 Check for read permission
- 2 Check for write permission
- 1 Execute (ignored)
- 0 Check for existence of file

int atoi(string szValue);

Convert the string to an integer value

int chdir(string szDir);

Change the current directory to the directory provided

int beep(void);

Make a sound using the built in PC speaker.

int date(int sYear, int sMonth, int sDay);

Returns the current year month and day as integer values.

int errno();

Returns the current errno value. This function can be used check to see if any of the file operations failed. If the return value is non zero this indicates that the last file I/O function call failed. See strerror() for more information on error handling.

int file_copy(string szFileSource, string szFileDestination);

Copy the source file to the given destination file.

int getcwd(string szDir);

Get the current directory

int getdisk();

Get the current disk number. The disk number is a:= 0, b: = 1, c:= 2, d: = 3 etc.

int getenv(string szVariable, string szResult);

Get the environment variable string and put the value into the string results buffer.

int remove(string szFileName);

Remove the specified file from the disk.

int rename(string szFileNameOld, string szFileNameNew);

Rename the old file to the new file name. Note that you can only rename files on the same disk drive.

int setdisk(int sDisk);

Set the new current disk number. The drive number is a:= 0, b: = 1, c:= 2, d: = 3 etc.

int sprintf(string szBuffer, string szFormat, [argument, ...]);

Generate a formatted string using the string format an the argument(s) provided and put the result into the buffer provided. Any combination of string, number or character arguments can be provided, up to a maximum of 6 in total.

The following format characters are supported:

%s - string format specifier
%d - integer format specifier
%c - character format specifier

int splitpath(string szPath, string szDrive, string szDir, string szFile, string szExtension);

The splitpath function will split the full path name into its components writing the results into the string return buffers provided. All five return buffers must be passed to splitpath but any of these buffers can be specified as null meaning that the corresponding component will be parsed but will not returned.

int strcat(string szString1, string szString2);

Concatenate the second string to the first string

int strchr(string szString1, char chTest);

Find the first occurrence of the chTest character in the string #1 and return the index position of that character. Returns -1 if the character was not found.

int strcmp(string szString1, string szString2);

Compare the two strings (return 0 if the same)

int strcpy(string szString1, string szString2);

Copy string #2 into string #1

int strerror(string szString1);

Returns a message in the string1 buffer corresponding to the last error. For more information also see the errno() function. Returns -1 if an error is encountered.

int strncpy(string szString1, string szString2, int sLength);

Copy sLength characters from string #2 into string #1

int stricmp(string szString1, string szString2);

Compare the two strings ignoring the case (return 0 if the same)

int strlen(string szString);

Get the length of the string

int strleft(string szString1, string szString2, int sLength);

Copy the left most sLength characters from string #2 and return the result in string #1

int strlwr(string szString);

Convert the string to lower case

int strupr(string szString);

Convert the string to upper case

int strmid(string szString1, string szString2, int sIndex, int sLength);

Copy the sLength characters from string #2 starting at position sIndex and return the result in string #1

int strrchr(string szString1, char chTest);

Find the last occurrence of the chTest character in the string #1 and return the index position of that character. Returns -1 if the character was not found.

int strright(string szString1, string szString2, int sLength);

Copy the right most sLength characters from string #2 and return the result in string #1

int strstr(string szString1, string szString2);

Find the first occurrence of the string #2 in the string #1 and return the index position of the string found. Returns -1 if the string was not found.

int time(int sHours, int sMinutes, int sSeconds, int sHundredths);

Returns the current time as an hour, minute, second and hundredths of a second as integer values.

Other Common Functions

The following function are common to both interpreters but in the case of the ZMI command line interpreter the output is to the screen while in the case of the Zeus Editor interpreter the output is sent to the currently active document.

int put_char(char chChar);

Puts a character to the output device.

int put_number(int sNumber);

Puts a number to the output device.

int put_string(string szFormat, [argument, ...]);

Puts a string to the output device. Any combination of string, number or character arguments can be provided, up to a maximum of 6 in total.

The following format characters are supported:

%s - string format specifier
%d - integer format specifier
%c - character format specifier

Zeus Editor Macro Functions

The following functions are only available to the interpreter built into the Zeus editor. They will generate runtime warnings if they are include in a script run by the ZMI command line interpreter. Note that any of these macros that deal with line or column positions expect these parameters to be zero based. Also remember that in addition to these built in macro functions it is also possible to include any of the Zeus TAG values or to run any of the built in keyboard functions by just entering the keyboard function name.

int beep_enable(int sEnable);

Enable or disable the Zeus sound feature. By default Zeus makes a sound each time an error or warning message is displayed on the status line. By passing in 0 or 1 this sound feature can be enabled or disabled as required.

int cursor_restore(void);

Restore the cursor to the position that was saved earlier using the *cursor_save* function.

int cursor_save(void);

Save the current cursor position in an internal buffer so that the cursor can be easily restored at a later time using the *cursor_restore* function. Note that this function is not recursive meaning the if this function is call on successive occasions the previously saved values are lost.

int getch([int sExtendedKey]);

Get a single character input from the keyboard. The return value represents the ASCII value of the key pressed. For information the case of non-ASCII keys the return value will be 0 and the key code information will be contained in the optional extended key data.

int get_char(char chChar);

Gets the character at the current cursor position in the current document.

int get_char_current(char chChar);

Gets the character at the current cursor position in the current document.

int get_char_next(char chChar);

Gets the character at the next cursor position in the current document.

int get_char_previous(char chChar);

Gets the character at the previous cursor position in the current document.

int get_column_pos(int sColumn);

Gets the cursor position in the current document.

int get_line(string szLineText, [int sColumn, int sLength]);

Gets a the text for the current line in the current document. The optional column and length values can be used to specify a starting column and a length. If not supplied they default to the first and last character respectively.

int get_line_count(int sCount);

Gets the total number of lines in the current document.

int get_line_pos(int sLine);

Gets the current line position.

int insert_line(string szLine);

Inserts the line of text into the current document.

int IsMarked(void)

Returns true if the current document has an area that is marked.

int is_document(void);

Returns 1 if the current active window is a document window. If the window is one of the output windows or no windows are open then this function will return a 0 value.

int locate_file(string szFileName);

Locates the full path name of the partial file name provided returning the result in the same string buffer. The search path used to located the file is the same as that used by the file search

built into the Zeus editor.

int message(string szFormat, [argument, ...]);

Displays a message on the status bar. Any combination of string, number or character arguments can be provided, up to a maximum of 6 in total.

The following format characters are supported:

%s - string format specifier
%d - integer format specifier
%c - character format specifier

int message_box(int sBoxType, string szMessage, [argument, ...]);

Display a message box of the specified type using the supplied message or build a message using the formatted message string the argument(s) provided.

The sBoxType (must be a value 1-5) determines the type of message box required. The following message boxes types are supported:

1 - Ok

2 - Ok and Cancel

3 - Ok, Cancel and Retry

4 - Yes and no

5 - Yes, No and Cancel

The formatted string can be any combination of string, number or character arguments up to a maximum of 6 in total. The following format characters are supported:

%s - string format specifier

%d - integer format specifier

%c - character format specifier

int open_file(string szFileName, int fReadOnly);

Opens the file specified with the specified read only mode. The read only flag will causes the file to be opened as read only mode an optional argument. By default the file will be opened with read/write access.

int open_file_name(string szFileName, [string szDirectory], [string szFilterName, string szFilterExt]);

Displays an open file dialog and returns the name of the file selected by the user. This function will return any empty string if the dialog is cancelled. The optional directory argument can be used to set the initial starting directory while the optional filter information can be used to configure an open dialog file filter. For example to get the name of a user selected text file from the project work directory the following code could be used:

```
string szFileName;
open_file_name(szFileName, "c:\project\work", "Text Files", "*.txt");
```

or if a multiple file extension filter is required then the following code could be used.

```
string szFileName;
open_file_name(szFileName, "c:\project\work", "C/C++ Files", "*.c;*.cpp;*.h;*.hpp");
```

int output_debug(string szFormat, [argument, ...]);

Displays a message to the standard output window. Any combination of string, number or character arguments can be provided, up to a maximum of 6 in total.

The following format characters are supported:

%s - string format specifier
%d - integer format specifier
%c - character format specifier

int read_file(string szFileName);

Reads a file into the current cursor position in the current document..

int search_restore(void);

Restore the search engine settings that where saved during the last call to the *search_save* function.

int search_save(void);

Save the current search engine settings. These can be later restored using a call to the *search_restore* function.

int search_update(int sEnabled);

By default the text searching functions automatically position the cursor to the text found. By passing in a 0 or 1 value this function allows this feature to be enabled or disabled respectively.

int set_column_pos(int sColumnPos);

Sets the cursor position in the current document.

int set_file_name(string szFileName);

Sets the name of the current document.

int set_find_text(string szText);

Sets the search text to be used for search/replace operations.

int set_line_pos(int sLine);

Sets the current line position in the current document.

int set_replace_text(string szText);

Sets the replace text to be used for search/replace operations.

int spawn(string szProgram, string szDirectory, int sFlags);

Start the specified program. The starting directory indicates the directory in which the program is run and the flags control the session. Both these arguments are optional. The starting directory can be 0 if not required and the control flags can be a "*bitwise OR"* of any of the following values:

1 - save the document before running the program

2 - capture any standard output generated by the program

4 - capture any standard error generated by the program

8 - ask for additional arguments

16 - the program will use the MS-DOS command interpreter (i.e. dir *.* etc)

32 - wait for the program to complete (the ESC key will cancel the wait)

64 - run the program in a visible DOS session

As an example. To capture all the output generated by the dir *.* command run in the in the c:\ temp directory you would use the following code:

```
int sFlags = 2 | 8 | 16;
```

```
spawn("dir *.*", "c:\temp", sFlags);
```

int user_input(string szPrompt, string szReply, string szTitle);

Get input from the user by displaying the szPrompt and reading the users reply into the szReply buffer. The szTitle string is an optional argument and if supplied it will be used to set the caption of the input dialog.

int user_input(string szPrompt1, string szReply1, string szPrompt2, string szReply2, string szTitle);

Get two inputs from the user by displaying the two szPrompt strings and reading the users reply into the two szReply buffers. The szTitle string is an optional argument and if supplied it will be used to set the caption of the input dialog.

int yield(void);

This function allows you to yield control back to windows, thus giving the impression of multi-tasking. For example if you have a macro that takes a long time to run, by just calling the yield() function periodically from within the macro will stop the system from appearing lock while the macro is running. But as a word of warning make sure you use this function wisely as writing a multi-tasking macro is a lot harder than writing a synchronous macro script.

ZMI Macro Functions

The following functions are only available to ZMI command line interpreter. They will generate runtime warnings if they are include in a script run by the Zeus interpreter.

int cls();

Clear the screen.

int file_to_stderr(string szFileName);

Send the specified file to standard output.

int file_to_stdout(string szFileName);

Send the specified file to standard error.

int getch();

Get a single character input from the standard input device (i.e. the keyboard).

int get_char(char chChar);

Get a character input from the user.

int get_number(int sNumber);

Get a number input from the user.

int get_string(string szString);

Get a string input from the user.

int printf(string szFormat, [argument, ...]);

Write the formatted string an the argument(s) provided to the standard output. Any combination of string, number or character arguments can be provided, up to a maximum of 6 in total.

The following format characters are supported:

- %s - string format specifier
- %d - integer format specifier
- %c - character format specifier

int spawn(string szProgram);

Start a specific program or run an MS-DOS command line command.

Running a Macro Script

There are two ways to run a Zeus macro script file. The first method is to use the ZMI.EXE file using the MS-DOS command line prompt. As an example run the ZMI.ZM script file which is located in the zScript directory. To run this script file just start an MS-DOS session and type in the following command:

```
zmi.exe zmi.zm
```

Alternatively just run the Macro Interpreter Example located on the Zeus Tools menu.

To run a macro script from inside the Zeus editor you need to use the Execute Macro Script dialog and supply the name of the macro script. You can use the Find option to help locate the script file that is to be run.

Macro Arguments

Although the macro function prototype suggests that command line arguments are not supported this is in fact is not the case. The Zeus macro language supports command line arguments through the use of the standard C like ***argc*** and ***argv*** variables. The ***argc*** and ***argv*** arguments are automatically defined and populated with the command line arguments during the macro initialisation. For a good example of how to use this command line argument feature refer to the examples provided.

Examples of Macro Scripts

Note that the Zeus macro system only supports one macro function per file and there is no provision for a macro to call another macro. So for all the macro examples shown below make sure that you copy each to its own macro file. All macros should be put in the zScript directory located in the Zeus install directory as this is default directory location used when Zeus goes looking for a given macro file.

ZMI.EXE Macro Examples

The following macro examples can only be run using the ZMI.EXE macro interpreter from any MS-DOS command line prompt or as a Zeus installed tool. To run any of these macros first copy the text to clipboard and then save it to a macro file (i.e. macro.zm). Once the file has been saved it can then be run from the MS-DOS command line prompt using the following command line: ZMI.EXE macro.zm

Zeus Macro Examples

The following macro examples can only be run using the Zeus built macro interpreter located in the macro commands menu item. To run any of these macros first copy the text to clipboard and then save it to a macro file (i.e. macro.zm). Then you can run the macro using an macro command menu item by just typing in the name of the macro file.

Incremental Search and Word Complete

For two more examples of some powerful scripts refer to the Incremental Search and Word Completion macros that have been attached to the Macros menu. Both scripts where development by Bertel K. Brander (bertel@post4.tele.dk) who needs to be congratulated for his fine work. These scripts show how easy it is to enhance functionality of the Zeus editor by using the built in scripting features provided. The source for both these script can be found in the WordComp.zm and ISearch.zm files while a short description of each of the scripts is given below.

```
       Name: WordComplete
Description: This macro provides a  word completion search feature.  To
             use this macro either run it directly or install it to the
             Macros menu using the Options Macros menu item. To use the
             macro just run it within or at the end of the current word
             that is to be completed. The following navigation keys are
             also supported:
                  ESC        - cancel the completion
                  Enter      - to accept the completed word
                  ArrowUp    - search for previous occurrence
                  ArrowDown  - search for next occurrence
             NOTE: Note that this macro does a case insensitive search.
     Author: Brander, Bertel K.
    Contact: bertel@post4.tele.dk

       Name: IncrementalSearch
Description: This macro provides an incremental search  feature.  To
             use this macro either run it directly or  install it to
             the Macros menu using the Options Macros menu item. The
             following keys are also supported:
                  ASCII Char - add the char to the search string
                  BackSpace  - remove last char from the search string
                  ESC        - cancel the search
                  Enter      - complete the search
                  ArrowUp    - search for previous occurrence
                  ArrowDown  - search for next occurrence
                  ArrowLeft  - same as backspace
                  ArrowRight - add the next char to the search string
     Author: Brander, Bertel K.
    Contact: bertel@post4.tele.dk
```

Hello Word (ZMI.EXE)

To run the following macro first save it to file then run the macro using the ZMI.EXE command line form an MS-DOS command line prompt.

```
//-- Name: HelloWorld
//-- Description: First macro script.
//-- Author: Jussi Jumppanen
int HelloWorld()
{
    //-- start with a clear screen
    cls();
    //-- print out the line of text
    printf("Hello world...\n");
    //-- not really needed but what the heck
    return 1;
}
```

Directory Functions (ZMI.EXE)

To run the following macro first save it to file then run the macro using the ZMI.EXE command line form an MS-DOS command line prompt.

```
//-- Name: DirMacro
//-- Description: Simple directory manipulation functions.
//-- Author: Jussi Jumppanen
int DirMacro(void)
{
    string strTest;
    getcwd(strTest);
    printf("Current Directory is: '%s'\n", strTest);
    sDisk = getdisk();
    printf("Current Disk is: %d\n", sDisk);
    // note:   a: = 0, b: = 1, c: = 2, d: = 3 etc
    setdisk(3);
    sDisk = getdisk();
    printf("New Disk is: %d\n", sDisk);
    string strTest1 = "\temp";
    chdir(strTest1);
    getcwd(strTest);
    printf("New Directory is: '%s'\n", strTest);
    printf("\n   Hit any key to continue.....\n");
    getch();
}
```

Command Line Arguments (ZMI.EXE)

To run the following macro first save it to file then run the macro using the ZMI.EXE command line form an MS-DOS command line prompt.

```
//-- Name: TestMacro
//-- Description: Command line arguments example.
//-- Author: Jussi Jumppanen
int TestMacro(void)
{
    //-- one way to get at the command line arguments
    string strTest = argv[0];
    printf("Argument #1 '%s'\n", strTest);
    strTest = argv[1];
    printf("Argument #2 '%s'\n", strTest);
    strTest = argv[2];
    printf("Argument #3 '%s'\n", strTest);
    strTest = argv[3];
    printf("Argument #4 '%s'\n", strTest);
    //-- another better way to get at the command line arguments
    for (i = 0; i < argc; ++i)
    {
        strTest = argv[i];
        printf("Argument #%d - '%s'\n", i, strTest);
    }
    printf("\n   Hit any key to continue.....\n");
    getch();
}
```

Hello Word (Zeus)

To run the following macro first save it to file then run the macro using the Macro Exectute Script dialog box.

```
//-- Name: HelloWorld
//-- Description: First macro script.
//-- Author: Jussi Jumppanen
int HelloWorld()
{
    //-- start with a new document (uses the FileNew keyboard function)
    FileNew();
    //-- give the file a name
    set_file_name("hello.txt");
    //-- print out the line of text (notice '\n' is now meaning less)
    put_string("Hello world...");
    //-- not really needed but what the heck
    return 1;
}
```

Command Line Arguments (Zeus)

To run the following macro first save it to file then run the macro using the Macro Exectute Script dialog box. Remember to also add some command line arguments to the execute script command.

```
//-- Name: TestMacro
//-- Description: Example of command line arguments.
//-- Author: Jussi Jumppanen
int TestMacro(void)
{
    FileNew();
    put_string("Argument Count: %d", argc);
    Enter();
    for (i = 0; i < argc; ++i)
    {
        sLength = strlen(argv[i]);
        put_string("Argument: %d : '%s' : Length %d", i, argv[i], sLength);
        Enter();
    }
}
```

Macro Tags (Zeus)

To run the following macro first save it to file then run the macro using the Macro Exectute Script dialog box.

```
//-- Name: TagExample
//-- Description: Example of using tag values
//-- Author: Jussi Jumppanen
int TagExample()
{
    //-- start with a new document (uses the FileNew keyboard function)
    FileNew();
    //-- the Zeus tags can be treated like string
    string szFile = "$FN";
    string szExt   = "$EB";
    put_string("File is '%s'               ", szFile);
    put_string("Extension is '%s' ", szExt);
}
```

Using the Yield Function (Zeus)

To run the following macro first save it to file then run the macro using the Macro Exectute Script dialog box.

```
//-- Name: YieldExample
//-- Description: Example of using the yield function
//-- Author: Jussi Jumppanen
int YieldExample()
{
    //-- start with a new document (uses the FileNew keyboard function)
    FileNew();
    put_string("Processing Started");
    for (i = 0; i < 30; ++i)
    {
        Enter();
        for (j = 0; j < 60; ++j)
        {
            //-- visual indication of processing running
            put_string(".");
            ScreenUpdate();
            for (k = 0; k < 1500; ++k)
            {
              //-- simulated processing delay loop
            }
            //-- make sure we keep windows running
            yield();
        }
    }
    Enter();
    put_string("Processing Complete");
}
```

Functional Comment Block

This macro will create a functional comment block. To run the following macro first save it to file then run the macro using the Macro Exectute Script dialog box.

```
//-- Name: FunctionBlock
//-- Description: Macro to create a functional block.
//-- Author: Jussi Jumppanen
int FunctionBlock()
{
  // only bother if it is a function
  if (is_document() == 0) return;
  string Author = "Jussi Jumppanen";
  string CurrentWord = "$W";
  if (strlen(CurrentWord) == 0)
  {
    // no current word so use the marked text if we have any
    CurrentWord = "$M";
  }
  // let the user has their say in the matter
  int Result = user_input("Function Name:", CurrentWord,
                          "                Author:", Author);
  if (Result)
  {
    int Day;
    int Year;
    int Month;
    date(Year, Month, Day);
    MoveLineHome();
    put_string("//-------------------------------------------------------------------------");
    EnterLine();
    put_string("//-- Function Name:    %s", CurrentWord);
    EnterLine();
    put_string("//--     Description:    ");
    cursor_save();
    put_string(".");
    EnterLine();
    put_string("//--            Author:    %s", Author);
    EnterLine();
```

```
      put_string("//-- Creation Date:    %d/%d/%d", Day, Month, Year);
      EnterLine();
      put_string("//-------------------------------------------------------------------------------");
      EnterLine();
      cursor_restore();
    }
    else
    {
      message("Operation was cancelled by the user");
    }
}
```

Redo

Redo the last undo command

ReplaceDialog

Display the Replace dialog

ReplaceForward

Display the Replace dialog with the search direction set to forward

ReplaceNext

Replace the next instance the search text as defined by the last run replace command

ReplacePrevious

Replace the previous instance the search text as defined by the last run replace command

ReplaceReverse

Display the Replace dialog with the search direction set to reverse

ReplaceWordCurrent

Display the Replace dialog with the search text set to match the current word provide the cursor is over a valid word

ScreenUpdate

Force a screen update in case the screen needs repainting (does not update commenting)

ScrollBarViewSet

Set the visible state of the document scroll bars to showing.

ScrollBarViewReset

Set the visible state of the document scroll bars to hidden.

ScrollBarViewToggle

Toggle the visible state of the document scroll bars.

ScrollLineDown

Scroll the currently active document down one line, maintaining the cursor at the same relative screen position

ScrollLineDownEx

Scroll the currently active document down one line but do not try to maintain the cursor at the same relative screen position

ScrollLineLeft

Scroll the currently active document left one character position

ScrollLinePageCenter

Scroll the current line to the center of the page

ScrollLinePageEnd

Scroll the current line to the end of the page

ScrollLinePageStart

Scroll the current line to the start of the page

ScrollLineRight

Scroll the currently active document right one character position

ScrollLineUp

Scroll the currently active document up one line, maintaining the cursor at the same relative screen position

ScrollLineUpEx

Scroll the currently active document up one line but do not try to maintain the cursor at the same relative screen position

ScrollLockReset

Turns the scroll locking feature off

ScrollLockSet

Turns the scroll locking feature on

ScrollLockToggle

Toggles the scroll locking feature on and off

SearchCaseReset

Turns the search case sensitivity off

SearchCaseSet

Turns the search case sensitivity on

SearchCaseToggle

Toggles the search case sensitivity on and off

SearchDialog

Display the search dialog

SearchDialogEx

Display the search dialog but will automatically dismiss the dialog after the find key is pressed.

SearchDirectionReset

Turns the search direction to forward

SearchDirectionSet

Turns the search direction to reverse

SearchDirectionToggle

Toggles the search direction between forward and reverse

SearchForward

Display the search dialog with the search direction set to forward

SearchForwardCount

Not yet implemented.

SearchNext

Repeat the last search in the forward direction

SearchPrevious

Repeat the last search in the reverse direction

SearchRegexpReset

Turns the search with regular expression off

SearchRegexpSet

Turns the search with regular expression on

SearchRegexpToggle

Toggles the search with regular expression on and off

SearchReverse

Display the search dialog with the search direction set to reverse

SearchReverseCount

Not yet implemented.

SearchWordCurrent

Display the Search dialog with the search text set to match the current word provide the cursor is over a valid word

SearchWordCurrentNext

Search for the next occurrence of the current word without displaying the search dialog

SearchWordCurrentPrevious

Search for the previous occurre of the current word without displaying the search dialog

SearchWordReset

Turns the search whole word option on

SearchWordSet

Turns the search whole word option off

SearchWordToggle

Toggles the search whole word option on and off

SoundError

Produce the error sound

SoundNote

Produce the note sound

SoundOk

Produce the OK sound

SoundQuestion

Produce the question sound

SoundWarning

Produce the warning sound

SpaceDelete

Not yet implemented

SpaceDeleteEnd

Not yet implemented

SpaceDeleteStart

Not yet implemented

SpellingDocument

Check the currently active document for spelling errors

SpellingWordCurrent

Check the current word for spelling errors

StampDay

Insert the numerical day value into the currently active document

StampMonth

Insert the numerical month value into the currently active document

StampYear

Insert the numerical year value into the currently active document

StampDateTime

Insert the full date/time stamp value into the currently active document

StampTime

Insert the time stamp value into the currently active document

StampFileName

Insert the name of current file into the currently active document

StampFileSize

Insert the size of current file into the currently active document

StampFileDateTime

Insert the date/time stamp when file was last modified into the currently active document

StandardOutputCopy

Copy the contents of the Standard Output Window to the clipboard

StandardOutputNext

Display the next entry from the standard output window

StandardOutputPrevious

Display the previous entry from the standard output window

StandardOutputView

Display the Standard Output Window

StatusBarViewReset

Reset the status bar display state. When the display state is set the status bar is visible

StatusBarViewSet

Set the status bar display state. When the display state is set the status bar is visible

StatusBarViewToggle

Toggles the status bar display state. When the display state is set the status bar is visible

Tab

If there is text that has been marked, move the marked text to the right by one tab character, else just insert a tab character

TabBack

If there is text that has been marked, move the marked text to the left by one tab character, else just move the cursor back one tab stop

TabBackChar

Move the cursor back one tab stop

TabChar

Insert a tab character

TabHard

Always insert a tab character irrespective of the current tabs as spaces editor option

TabSoft

Always insert a tab character as spaces irrespective of the current tabs as spaces editor option

ToolBarViewReset

Reset the toolbar display state. When the display state is set the tool bar is visible

ToolBarViewSet

Set the toolbar display state. When the display state is set the tool bar is visible

ToolBarViewToggle

Toggles the tool bar display state. When the display state is set the tool bar is visible

ToolsCommand

Display the DOS Command Line dialog

ToolsExecute1

Run the currently install tool number 1. If no tool has been install this command has no effect

ToolsExecute2

Run the currently install tool number 2. If no tool has been install this command has no effect

ToolsExecute3

Run the currently install tool number 3. If no tool has been install this command has no effect

ToolsExecute4

Run the currently install tool number 4. If no tool has been install this command has no effect

ToolsExecute5

Run the currently install tool number 5. If no tool has been install this command has no effect

ToolsExecute6

Run the currently install tool number 6. If no tool has been install this command has no effect

ToolsExecute7

Run the currently install tool number 7. If no tool has been install this command has no effect

ToolsExecute8

Run the currently install tool number 8. If no tool has been install this command has no effect

ToolsExecute9

Run the currently install tool number 9. If no tool has been install this command has no effect

ToolsExecute10

Run the currently install tool number 10. If no tool has been install this command has no effect

ToolsExecute11

Run the currently install tool number 11. If no tool has been install this command has no effect

ToolsExecute12

Run the currently install tool number 12. If no tool has been install this command has no effect

ToolsExecute13

Run the currently install tool number 13. If no tool has been install this command has no effect

ToolsExecute14

Run the currently install tool number 14. If no tool has been install this command has no effect

ToolsExecute15

Run the currently install tool number 15. If no tool has been install this command has no effect

ToolsShell

Spawn a DOS command line session

Undo

Undo the last made change

UndoFlush

Flush the undo buffer for the currently active document

Version

Display the current version of the software

WindowArrange

Arrange the currently active MDI windows

WindowCascade

Cascade the currently active MDI windows

WindowCloseAll

Close all the currently active MDI windows, making sure all changes have been saved

WindowMaximizeAll

Maximize all the currently open windows

WindowMinimizeAll

Minimize all the currently open windows

WindowRestoreAll

Restore the size of all the currently open windows

WindowNext

Move to the next active MDI window

WindowPrevious

Move to the previous active MDI window

WordTextReverse

Perform a text reversal operation on the current word

WindowTile

Tile the currently active MDI windows

WindowTileHorizontal

Tile the currently active MDI windows with a horizontal aspect

WindowTileVertical

Tile the currently active MDI windows with a vertical aspect

WordCaseLower

Convert the current word to lower case

WordCaseTranspose

Convert the current word lower case characters to upper case and vice versa

WordCaseUpper

Convert the current word to upper case

WordDelete

Delete the current word

WordDeleteEnd

Delete from the current position to the end of the current word

WordDeleteNext

Delete the next word

WordDeletePrevious

Delete the previous word

WordDeleteStart

Delete from the start of the word up to the current position

WordCopy

Copy the current word to clipboard

WordCut

Cut the current word to clipboard

WordDeleteNextEx

Delete the next word or join the line with the next line if there is no next word to delete

WordDeletePreviousEx

Delete the previous word or join the line with the previous line if there is no previous word to delete

Backspace

Moves the cursor back one position deleting the previous character

BackspaceEx

Moves the cursor back one position deleting the previous character and if you reach the start of the current line move the cursor to the end of the previous line

BackspaceSmart

Perform a backspace to a column position using the line above as a tab stop template.

FileOpenEx

Open a file from disk, making it the currently active document but use the directory of the currently active file as the starting directory of the open dialog.

HelpSupport

Display information on how to get technical support for this software.

SortAscending

Sort the file in ascending order.

SortDescending

Sort the file in descending order.

FileAnsiToOem

Translates the current document into the OEM-defined character set.

FileOemToAnsi

Translates the current document from the OEM-defined character set into either an ANSI or a wide-character string.

TabSmart

Perform a tab to a column position using the line above as a tab stop template.

TabBackSmart

Perform a back tab to a column position using the line above as a tab stop template.

SortMarkedAscending

Sort the marked lines in ascending alphabetical order. Note that this function will always sort the entire line but the actual sorting keys are defined by the marked area. This means that if a portion of a line is marked with column marking the range of lines marked will be sorted using the text select which is not necessarily the same as entire line of text text.

SortMarkedDescending

Sort the marked lines in descending alphabetical order. Note that this function will always sort the entire line but the actual sorting keys are defined by the marked area. This means that if a portion of a line is marked with column marking the range of lines marked will be sorted using the text select which is not necessarily the same as entire line of text text.

BookMarkDrop0

Save the current line number against bookmark number 0

BookMarkDrop1

Save the current line number against bookmark number 1

BookMarkDrop2

Save the current line number against bookmark number 2

BookMarkDrop3

Save the current line number against bookmark number 3

BookMarkDrop4

Save the current line number against bookmark number 4

BookMarkDrop5

Save the current line number against bookmark number 5

BookMarkDrop6

Save the current line number against bookmark number 6

BookMarkDrop7

Save the current line number against bookmark number 7

BookMarkDrop8

Save the current line number against bookmark number 8

BookMarkDrop9

Save the current line number against bookmark number 9

BookMarkGoto

Move the cursor to the line stored against the bookmark selected. A valid bookmark is any number between 0 and 9.

BookMarkGoto0

Move the cursor to the line stored against the bookmark number 0

BookMarkGoto1

Move the cursor to the line stored against the bookmark number 1

BookMarkGoto2

Move the cursor to the line stored against the bookmark number 2

BookMarkGoto3

Move the cursor to the line stored against the bookmark number 3

BookMarkGoto4

Move the cursor to the line stored against the bookmark number 4

BookMarkGoto5

Move the cursor to the line stored against the bookmark number 5

BookMarkGoto6

Move the cursor to the line stored against the bookmark number 6

BookMarkGoto7

Move the cursor to the line stored against the bookmark number 7

BookMarkGoto8

Move the cursor to the line stored against the bookmark number 8

BookMarkGoto9

Move the cursor to the line stored against the bookmark number 9

BraceMatch

Finds the matching brace for the brace character a the current cursor position. A valid brace character is one of the following characters:

"[]{}<>()"

BraceMatchForward

Finds the matching reverse brace for the forward brace character a the current cursor position. A valid forward brace character is one of the following characters:

"[{<("

BraceMatchReverse

Finds the matching forward brace for the reverse brace character a the current cursor position. A valid reverse brace character is one of the following characters:

"]}>)"

BraceMatchForwardEx

Search for the next brace charcater in a forward direction.

BraceMatchReverseEx

Search for the next brace charcater in a reverse direction.

CharCopyFromLineAbove

Copies a character to the current line based on the corresponding character at the same cursor position in the line above.

CharCopyFromLineBelow

Copies a character to the current line based on the corresponding character at the same cursor position in the line below

CharDelete

Deletes the character at the current cursor position

CharQuote

Causes the next character to be enter to be treated literally even if it is a command keystroke

CharSwapNext

Swap the current character with the next character

CharSwapPrevious

Swap the current character with the previous character

ClipboardPaste

Paste the contents of the clipboard into the current cursor position

ClipboardPasteAndMark

Paste the contents of the clipboard into the current cursor position and mark the text that was just added

CompilerCompile

Compiles the currently active document

CompilerOutputCopy

Copy the contents of the Compiler Output Window to the clipboard

CompilerOutputNext

Moves the to the next compiler output or warning, be it in the compiled document or in the output window itself

CompilerOutputPrevious

Moves the to the previous compiler output or warning, be it in the compiled document or in the output window itself

CompilerOutputView

Display the Compiler Output Window

CompilerSetup

Display the compiler setup dialog

Enter

Causes a new line to be enter with the cursor moving to the new line

EnterNext

Causes a new line to be enter without splitting the current line, with the cursor moving to the new line

EnterOpen

Causes a the current line to be split but the cursor position is maintained at its current location

EnterLine

Causes a new line to be enter with the cursor moving to the new line but do not do any smart indenting or smart brace processing

FileBackupReset

Turns the automatic file backup option off

FileBackupSet

Turns the automatic file backup option on

FileBackupToggle

Toggles the automatic file backup option on and off

FileBackupWrite

Forces a backup write for the current active document

FileClose

Close the currently active document

FileExit

Close the application, checking that all documents have been saved

FileInsertAtCursor

Insert a file into the current active document at the current line number

FileListAssociates

Display a list of all the files that share the same base file name as the currently active document. Any associated file can then be loaded by just selecting the required file from this list

FileListDisplay

Display a list of all the currently active document and output windows

FileListDisplayEx

Display a list of all the currently active document and output windows but use an improved dialog box.

FileName

Display the name of the currently active document

FileNew

Create a new untitled document, making it the currently active document

FileOpen

Open a file from disk, making it the currently active document

FileOpenInLine

Try to open the file as described by the text of the current line. This can be used to open include files

FilePrint

Print the current active document

FilePrintAll

Print all the currently open documents

FilePrintSetup

Display the Print Setup dialog, thus allowing the printer to be configured

FileReadOnlyModeReset

Reset the file open read only mode. The read only mode determines the mode all subsequent files are opened

FileReadOnlyModeSet

Set the file open read only mode. The read only mode determines the mode all subsequent files are opened

FileReadOnlyModeToggle

Toggles the file open in read only mode. The read only mode determines the mode all subsequent files are opened

FileReadOnlyReset

Reset the read only status of the current file. If the read only mode is set the file cannot be modified

FileReadOnlySet

Set the read only status of the current file. If the read only mode is set the file cannot be modified

FileReadOnlyToggle

Toggles the read only status of the current file. If the read only mode is set the file cannot be modified

FileReadOEMReset

Turns the File Read as OEM option off

FileReadOEMSet

Turns the File Read as OEM option on

FileReadOEMToggle

Toggles the File Read as OEM option on and off

FileReload1

Reload the most recently open document number 1

FileReload2

Reload the most recently open document number 2

FileReload3

Reload the most recently open document number 3

FileReload4

Reload the most recently open document number 4

FileReload5

Reload the most recently open document number 5

FileReload6

Reload the most recently open document number 6

FileReload7

Reload the most recently open document number 7

FileReload8

Reload the most recently open document number 8

FileReload9

Reload the most recently open document number 9

FileReloadCurrent

Reload the currently active document from disk

FileSave

Save the currently active document

FileSaveAll

Save all the currently active documents

FileSaveAllNamed

Save all the currently active documents that are named. This means untitled documents will not be saved

FileSaveAs

Save the currently active document under a different name

FileTouch

Force's the time stamp of the currently active file to be touched

FileWriteOEMReset

Turns the File Write as OEM option off

FileWriteOEMSet

Turns the File Write as OEM option on

FileWriteOEMToggle

Toggles the File Write as OEM option on and off

FunctionFindAll

Find all the function definitions for the currently active document

FunctionFindNext

Find the next function definition for the currently active document

FunctionFindPrevious

Find the previous function definition for the currently active document

HelpAbout

Display the help about dialog box

HelpIndex

Display the online help index information

HelpKeys

Display the online help on keys information

HelpQuickHelp

Perform a quick help search on the current word to the currently selected text. This will perform a quick help keyword search of the online help files installed

HelpQuickSearch

Display the Quick Search dialog. This will allow you to perform a quick help keyword search of the online help files installed

HelpQuickSetup

Display the Quick Configuration dialog. This will allow you to install online help files that are to be searched by the quick help search engine

HelpRegister

Display the shareware registration information

HelpUsing

Display the help on using the online help facility

InsertModeReset

Turns the character insert mode off

InsertModeSet

Turns the character insert mode on

InsertModeToggle

Toggles the character insert mode on and off

InsertSpace

Insert a single space character but retain the cursor position

IsEndOfLine

Returns true if the cursor is at the end of the current line

IsOutsideLine

Returns true if the cursor is past the end of the current line

IsStartOfLine

Returns true if the cursor is at the start of the current line

IsWithinLine

Returns true if the cursor is at within the current line

LineCaseLower

Convert the current line to lower case

LineCaseTranspose

Convert the current line upper case characters to lower case and vice versa

LineCaseUpper

Convert the current line to upper case

LineCopy

Copy the current line to the clipboard

LineCopyAppend

Copy the current line and append the result to the clipboard.

LineCopyEnd

Copy the text from the current cursor to the end of the line to the clipboard

LineCopyEndAppend

Copy from the current cursor to the end of the line and append the result to the clipboard.

LineCut

Cut the current line to the clipboard

LineCutAppend

Cut the current line and append the result to the clipboard.

LineCutEnd

Cut the text from the current cursor to the end of the line to the clipboard

LineCutEndAppend

Cut from the current cursor to the end of the line and append the result to the clipboard.

LineCutEndEx

Cut the text from the current cursor to the end of the line to the clipboard or join the line with the line below if the cursor is currently at the end of the line

LineCutStart

Cut the text from the current cursor to the start of the line to the clipboard

LineDelete

Delete the current line. The line is not added to the clipboard

LineDeleteEnd

Delete all the characters in the current line starting form the current position up to the end of the line

LineDeleteEndEx

Clear the text from the current cursor position to the end of the line or join the line with the line below if the cursor is currently at the end of the line

LineDeleteStart

Delete all the characters in the current line between the current position and the start of the line

LineGoto

Display the line goto dialog box. From here you can enter the line number to which the cursor should be moved

LineJoinNext

Join the current line with the next line

LineJoinPrevious

Join the current line with the previous line

LineTextReverse

Reverse the order of all the text contained in the current line

LineWrap

Perform a check for line wrap on the current line and if it is found to be too long it will be wrapped. In the case that the current line does get wrapped this function does not check to see if the resulting new line also needs to be wrapped

LineWrapEx

Perform a check for line wrap on the current line and if it is found to be too long it will be wrapped. In the case that the current line does get wrapped this function also checks to see if the resulting new lines also need to be wrapped and if they do this wrapping will also be performed

LineWrapReset

Turns the automatic line wrap feature off

LineWrapSet

Turns the automatic line wrap feature on

LineWrapToggle

Toggles the automatic line wrap feature on and off

LineWrapMarkedArea

Force the current marked area to be check for possible line wrapping

MacroExecute1

Run the currently install macro number 1. If no macro has been install this command has no effect

MacroExecute2

Run the currently install macro number 2. If no macro has been install this command has no effect

MacroExecute3

Run the currently install macro number 3. If no macro has been install this command has no effect

MacroExecute4

Run the currently install macro number 4. If no macro has been install this command has no effect

MacroExecute5

Run the currently install macro number 5. If no macro has been install this command has no effect

MacroExecute6

Run the currently install macro number 6. If no macro has been install this command has no effect

MacroExecute7

Run the currently install macro number 7. If no macro has been install this command has no effect

MacroExecute8

Run the currently install macro number 8. If no macro has been install this command has no effect

MacroExecute9

Run the currently install macro number 9. If no macro has been install this command has no effect

MacroExecute10

Run the currently install macro number 10. If no macro has been install this command has no effect

MacroExecute11

Run the currently install macro number 11. If no macro has been install this command has no effect

MacroExecute12

Run the currently install macro number 12. If no macro has been install this command has no effect

MacroExecute13

Run the currently install macro number 13. If no macro has been install this command has no effect

MacroExecute14

Run the currently install macro number 14. If no macro has been install this command has no effect

MacroExecute15

Run the currently install macro number 15. If no macro has been install this command has no effect

MacroExecuteScript

Display the macro execute dialog which allows you to load and execute a macro script file

MacroLoad

Load a macro from file, making it the current macro

MacroPlay

Play the current macro, be it recorded or loaded from disk

MacroRecordReset

Reset the macro recording state. When the macro recording state is set all keyboard keystrokes are recorded against the current macro

MacroRecordSet

Set the macro recording state. When the macro recording state is set all keyboard keystrokes are recorded against the current macro

MacroRecordToggle

Toggles the macro recording state. When the macro recording state is set all keyboard keystrokes are recorded against the current macro

MacroRepeat

Display the macro repeat dialog so that the current macro can be played a repeated number of times

MacroSave

Save the current macro to a macro file

MarkBlockReset

Turns the block marking mode off

MarkBlockSet

Turns the block marking mode on

MarkBlockSetEx

Turns the block marking mode on but does not remove any previous mark that may have been active

MarkBlockToggle

Toggles the block marking mode on and off

MarkBlockToggleEx

Toggles the block marking mode on or off but does not remove any existing marked area.

MarkCaseLower

Convert the marked text to lower case

MarkCaseTranspose

Convert the marked text lower case characters to upper case and vice versa

MarkCaseUpper

Convert the marked text to upper case

MarkColumnReset

Turns the column marking mode off

MarkColumnSet

Turns the column marking mode on

MarkColumnSetEx

Turns the column marking mode on but does not remove any previous mark that may have been active

MarkColumnToggle

Toggles the column marking mode on and off

MarkColumnToggleEx

Toggles the column marking mode on or off but does not remove any existing marked area.

MarkCopy

Copy the marked area to clipboard

MarkCopyEx

Copy the marked area or the current line to clipboard

MarkCopyToCursor

Copy the marked area to the current cursor location

MarkCopyToCursorEx

Copy the marked area to the current cursor location but retain the marked area after the copy is complete

MarkCursorEnd

Move the cursor to the end of the marked area

MarkCursorStart

Move the cursor to the start of the marked area

MarkCursorToggle

Toggles the cursor position between the beginning and end of the currently marked area

MarkCut

Cut the marked area to clipboard

MarkCutEx

Cut the marked area or the current line to clipboard

MarkDelete

Delete the marked area. The text deleted is not added to the clipboard

MarkDeleteEx

Delete the marked area or the current line. The text deleted is not added to the clipboard

MarkFillWithChar

Not yet implemented

MarkFillWithSpace

Fill the currently marked area with space characters. This function will add characters past the end of line if the line is so marked.

MarkFillWithSpaceEx

Fill the currently marked area with space characters. This function will not add characters past the end of line.

MarkHide

Remove the current marked area, leaving the marked text unchanged

MarkLineReset

Turns the line marking mode on

MarkLineSet

Turns the line marking mode off

MarkLineSetEx

Turns the line marking mode on but does not remove any previous mark that may have been active

MarkLineToggle

Toggles the line marking mode on and off

MarkLineToggleEx

Toggles the line marking mode on or off but does not remove any existing marked area.

MarkMoveToCursor

Move the marked area to the current cursor location

MarkMoveToCursorEx

Move the marked area to the current cursor location but retain the marked area after the move is complete

MarkPaste

Paste the contents of the clipboard, replacing any text that has been marked

MarkPasteEx

Paste the contents of the clipboard, replacing any text that has been marked, or just insert the clipboard data if no area has been marked (Brief like paste operation)

MarkPrint

Print the currently marked area

MarkSelectAll

Mark the entire contents of the currently active document leaving the document in marking mode

MarkSelectAllEx

Mark the entire contents of the currently active document but turn of the marking once the document text has been selected

MarkShiftLeft

Shift the currently marked area one tab space the left

MarkShiftRight

Shift the currently marked area one tab space the right

MarkShiftLeftEx

Shift the currently marked area to the previous tab stop

MarkShiftRightEx

Shift the currently marked area to the next tab stop

MarkTextReverse

Not yet implemented

MarkWordCurrent

Mark the word under the current cursor location

MarkWordEnd

Mark from the current cursor location to the end of the current word

MarkWordStart

Mark from the start of the current word up to the current cursor location

MarkWriteToFile

Write the currently marked area to file

MoveDocumentCenter

Move the cursor to the center of the currently active document

MoveDocumentEnd

Move the cursor to the end of the currently active document

MoveDocumentEndEx

Move the cursor of the currently active document catering for the special Brief navigation sequence, end of line, end of page and end of document

MoveDocumentStart

Move the cursor to the start of the currently active document

MoveDocumentStartEx

Move the cursor of the currently active document catering for the special Brief navigation sequence, start of line, start of page and start of document

MoveLineCenter

Move the cursor to the center of the current line

MoveLineDown

Move the cursor down one line position

MoveLineDownAndFirst

Move the cursor down to the next line and position the cursor at the first non-white space character of that line

MoveLineDownAndLast

Move the cursor down to the next line and position the cursor at the first non-white space character of that line

MoveLineEnd

Move the cursor to the end of the line

MoveLineFirst

Move the cursor to the first non-white character in the current line

MoveLineHome

Move the cursor to the start of the line

MoveLineLeft

Move the cursor one character position to the left

MoveLineLeftEx

Move the cursor one character position to the left but reposition the cursor to the end of the previous line if you reach the start of the current line

MoveLineLeftEdge

Move the cursor to the character at the left edge of the screen

MoveLineRight

Move the cursor one character position to the right

MoveLineRightEx

Move the cursor one character position to the right but reposition the cursor to the start of the next line you reach the end of the current line

MoveLineRightEdge

Move the cursor to the character at the right edge of the screen

MoveLineUp

Move the cursor up one line position

MoveLineUpAndFirst

Move the cursor up to the previous line and position the cursor at the first non-white space character of that line

MoveLineUpAndLast

Move the cursor up to the previous line and position the cursor at the last non-white space character of that line

MovePageCenter

Move the cursor to the center of the current page

MovePageDown

Move the cursor down one page

MovePageEnd

Move the cursor to the end of the current page

MovePageLeft

Move the cursor to the left one page

MovePageRight

Move the cursor to the right one page

MovePageStart

Move the cursor to the start of the current page

MovePageUp

Move the cursor up one page

MoveWordCenter

Move the cursor to the center of the current word

MoveWordEnd

Move the cursor to the end of the current word

MoveWordNext

Move the cursor to the start of the next word

MoveWordNextEx

Move to the next word and continue onto the next line if you reach the end of the current line

MoveWordPrevious

Move the cursor to the start of the previous word

MoveWordPreviousEx

Move to the previous word and continue onto the previous line if you reach the beginning of the current line

MoveWordStart

Move the cursor to the start of the current word

NotSupported

Display a message saying 'This keystroke is not supported'

OptionsColors

Display the Color Options dialog

OptionsEditor

Display the Editor Options dialog

OptionsElectric

Display the Templates Options dialog

OptionsExtension

Display the Extensions Options dialog

OptionsFilters

Display the Filter Options dialog

OptionsFont

Display the Font dialog

OptionsKeyboard

Display the Keyboard Options dialog

OptionsMacros

Display the Macro Options dialog

OptionsSpelling

Display the Spelling Options dialog box

OptionsTools

Display the Tool Options dialog

PopupMenuDisplay

Display the popup edit menu list

ProjectClose

Close the currently open project

ProjectCurrent

Display the name of the currently open project

ProjectDisplay

Display the file that make up the currently open project

ProjectExecute

Not yet implemented

ProjectExecuteDebug

Execute the debug command line as defined in the projects settings dialog

ProjectMake

Make the currently open project

ProjectMakeAll

Make all the components of the currently open project

ProjectOpen

Open a project file

ProjectOutputCopy

Copy the contents of the Project Output Window to the clipboard

ProjectOutputNext

Display the next entry from the project output window

ProjectOutputPrevious

Display the previous entry from the standard output window

ProjectOutputView

Display the Project Output Window

ProjectSetup

Display the Project Options dialog

TagsSetup

Display the tag setup dialog.

TagsBuild

Build the tag file using the build tag command.

TagsDisplayNext

Display the next tag found.

TagsDisplayPrevious

Display the previous tag found.

TagsDisplayResults

Display the all the matching tags found.

TagsDisplayTagFile

Display the contents of the tag file.

TagsFindDialog

Display the Search Tag dialog.

TagsFindWordCurrent

Find the current word in the tag file.

TagsFindWordCurrent1

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsFindWordCurrent2

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsFindWordCurrent3

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsFindWordCurrent4

Find the current word in the tag file but refine the search using the extra tag field information as set in the tag setup dialog.

TagsDisplayUndo

Undo the last tag display operation.

TagsDisplayRedo

Redo the last tag display undo operation.

Build Command

Enter the typical command line needed to build a tag file called SAMPLE.TAG. To check the command line entered try using the same command line from any DOS session.

Tag File

This is the name and location of the tag file to be produced. This name will be used as a substitute for the SAMPLE.TAG value defined in the build command.

Match case sensitive

Use this option to make the tag search either case sensitive or case insensitive.

Ask for rebuild tag file if the tag is not found

Use this option to force a prompt for tag file rebuild if the tag search fails to find the requested tag.

Help debug tag building

Use this option to help debug the tag build process. If the tag build is not producing a tag file this option can be used to provide a more detail regarding the build cycle and this may help determine the source of the build problem.

Tag Field

Use these fields to specify the tag field data for the corresponding TagsFindWordCurrent# keyboard function. These fields allow special search data to be defined, which is used as an extra search filter when searching the tags file. This field can be any valid regular expression. Note that these fields will only work with the new tag file format. For more details on how to use this field, refer to the "Understanding the Tags Support" for more details.

Tags Options Dialog

The Tags Options dialog is used to configure the Zeus tag support. Once you have configured the tag options it then becomes possible to use the Tags menu functions or the built in tag keyboard functions. For more information on each of the different sections of this dialog click on one of the items listed below.

Build Command

Tag File

Match case sensitive

Ask for rebuild tag file if the tag is not found

Help debug tag building

Tag Field #1

Tag Field #2

Tag Field #3

Tag Field #4

OK

Cancel

If you have any problems getting the tool to run correctly refer to the Understanding the Tags Support section for more details. Also remember that the you can also include one of the many Zeus TAG macros in the build command line and tag file name entry fields.

Understanding Tags Support

Zeus allows you to use third party tagging tools to further enhance the functionality of Zeus editor. The Zeus tag support can be configured using the Tags Options dialog. This section describes how the Zeus tag support is designed to work and provides examples on how the to configure Zeus to use external tag tools.

Finding a Suitable Tag Tool

The Zeus install will have already installed a simple ctags.exe tag tool and by default the Zeus tags setup will be automatically configured to use this tool. Although this tool is useful as a tutorial on how to use tags, in reality a more powerful tag tool should be is used. As an example of one such tool:

Zeus Tag Support

The Zeus tag support is designed to conform to the new tag file format, as first proposed by Darren Hiebert <darren@hiebert.com>. This new format looks something like this:

```
      {tagname}<Tab>{tagfile}<Tab>{tagaddress}[;"<Tab>{tagfield}..]

   Standard Fields:

   {tagname}    Any identifier, not containing white space.
   <Tab>        Exactly one TAB character.
   {tagfile}    The name of the file where {tagname} is defined,
relative
                to the current directory (or location of the tags
file?).
   {tagaddress} Any Ex command.  When executed, it behaves like 'magic'
                was not set.  It may be restricted to a line number or a
                search pattern (Posix).

   Optionally Fields:

   ;"           semicolon + double quote:
   {tagfield}   A tagfield is program specific and consists of a name, a
                colon, and a value for example: "name:value".
```

For more information regarding this new format and for information on downloading an extremely powerful tag generator program that will produce tag files using this new format, visit the following web page (or search the internet for "Exuberant Ctags"):

```
http://fly.hiwaay.net/~darren
```

The Zeus tag support has been enhanced to support this new tag file format, but support for the older tag file format is also supported. In its most basic form the Zeus tag support should work with almost any tag generator, provided the tag file produced has the tag name as it's first element and it is delimited by a tab character. For example the most basic tag format supported would be:

```
tag_name<tab>anything you like can be here
```

The Zeus behaviour with regard to the new tag file format changes depending on which of the Zeus tag function is used. These differences in behaviour are described below:

TagsFindDialog

This function allows for a search on both the tag name and the tag field. For a tag to be considered as a match both the tag name and tag field must be found. If no tag field is supplied then only the tag name is used to find the matching the tag. The tag field can be any valid regular expression.

TagsFindWordCurrent

This function will try to find a matching tag using only the tag name value. In this case the tag field value is not considered. The tag name to found is automatically derived as the current word of the currently active document.

TagsFindWordCurrent1

TagsFindWordCurrent2

TagsFindWordCurrent3

TagsFindWordCurrent4

These functions will try to find a matching tag using the tag name and the tag field values. The tag name to be found is automatically derived as the current word of the currently active document and the tag field corresponds to the tag field text as previously entered into the tag field section of the tags setup dialog box. This tag field can be any valid regular expression.

Examples of Tag Configurations

Note that the example shown below uses the built in Zeus macro tags. For example $ZD equates to the zeus install directory and $FDD represents the file drive and directory of the current active document.

Example #1:

Using ctags.exe as supplied by Zeus use the options tag setup menu to configure the tags as follows:

```
  Build Command: $ZDctags.exe -stca -fSAMPLE.TAG $FDD*.c
       Tag File: $ZDzeus.tag
    Tag Field#1: Not supported by ctags.exe
    Tag Field#2: Not supported by ctags.exe
    Tag Field#3: Not supported by ctags.exe
    Tag Field#4: Not supported by ctags.exe
  Other Options: Set help debug to on and case sensitive to on.
```

Using this example, the build tag process will run ctags to produce the 'zeus.tag' tags file in the Zeus install directory. The files that will be used to produce the tags will be based on all the *.c and *.h files in the directory of the currently loaded document. The Tags - Tag Current Word menu item can then be used to locate the current word using this tag file or the actual tag file can be displayed using the Tags - Display Tags File menu option.

Example #2:

This example assumes you have downloaded the ctags program written by Darren Hiebert (see the http://fly.hiwaay.net/~darren web page for details). Since this tag program produces a tag file using the new tag format, this allows the tag field values to be used. One of the features of this tag program is that it produces the tagfield information in the tag file output. The tag field's produced can include the following additional data:

- c class names
- d macro definitions
- e enumerators (values inside an enumeration)
- f function (or method) definitions
- g enumeration names
- m class, structure, and union data members
- p external function prototypes
- s structure names
- t typedefs
- u union names

v variable declarations

Using this ctags.exe use the options tag setup menu to configure the tags as follows:

```
Build Command: ctags.exe -a -fSAMPLE.TAG
     Tag File: $ZDzeus.tag
  Tag Field#1: [ \t]+c[^a-zA-Z]+
  Tag Field#2: [ \t]+d[^a-zA-Z]+
  Tag Field#3: [ \t]+v[^a-zA-Z]+
  Tag Field#4: [ \t]+f[^a-zA-Z]+
Other Options:  Set help debug to on and case sensitive to on.
```

This example will work identically to the first example but in this case the tag field values can now be defined. In this case we have define the tag fields to map to class names, macro definitions, variable or functions. These fields map to there corresponding TagsFindWordCurrent# keyboard functions (where # is the number 1-4). Thus, once the tag file has been built, place the cursor on a word and run the TagsFindWordCurrent4 keyboard function. Zeus will then search the tag file for the current word but will only accept the tag as found if the word exists in the tag file and if the tag is in fact a function definition. Only when both these conditions is the tag accepted as a match. When the tag fields are left blank they are ignored.

Using this feature it is now possible to configure the TagsFindWordCurrent# keyboard to only search a particular type of tag, for example function definitions, structures, typedefs, class declarations etc etc.