



EditPad Pro

Manual

Version 4.5.4 – 12 February 2003

Copyright © 1996–2003 Jan Goyvaerts. All rights reserved.
"EditPad" and "JGsoft – Just Great Software" are trademarks of Jan Goyvaerts

Welcome to the EditPad Pro Manual

Welcome to the documentation for EditPad Pro. You will find that this documentation contains elaborate descriptions of each feature that EditPad Pro offers, so that even people with little or no experience with text editors like EditPad can understand. People with plenty of experience may find this unnecessary. However, these people can probably use EditPad without any documentation at all.

If you are using EditPad for the first time, you may want to start with configuring it to your own preferences.

After explaining the various settings you can make in Options|Preferences, this manual will explain each feature of EditPad Pro menu item by menu item.

Visit <http://www.editpadpro.com/> for the latest information on EditPad Pro.

Send an email to support@editpadpro.com for EditPad Pro and support@editpadlite.com for EditPad Lite if you need technical support or have some comments you want to share.

If you have purchased EditPad Pro, you are entitled to free technical support by email. If you are using EditPad Lite, we do appreciate your input, but we cannot promise a personal reply.

Please use Help|Check for New Version to see if you are using the latest version, before asking for technical support.



Contents

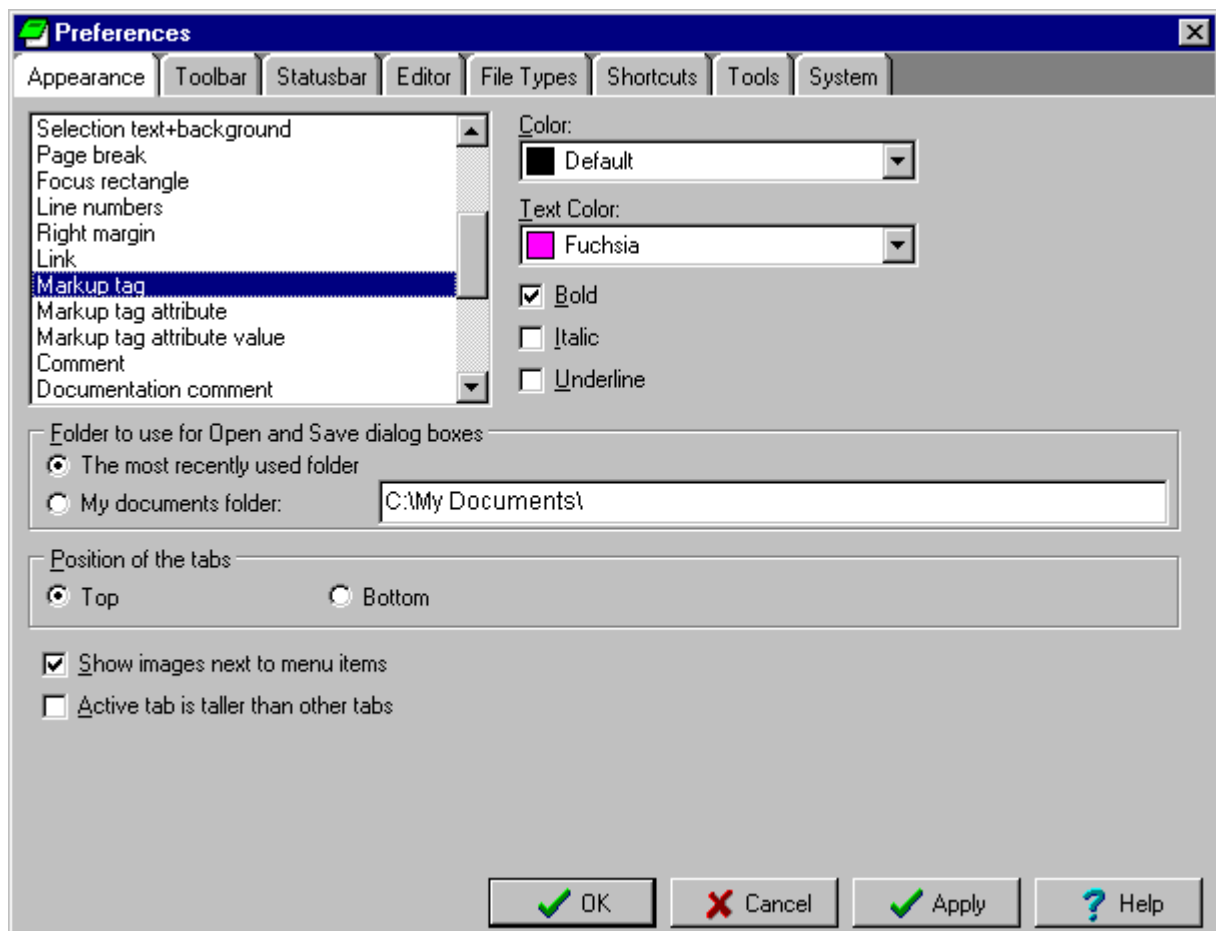
Preferences	5	About EditPad Pro Projects	37
Appearance Preferences.....	5	Project Open Project.....	37
Toolbar Preferences	7	Project Save Project	37
Statusbar Preferences.....	9	Project Reopen	37
Editor Preferences.....	10	Block Indent	38
File Types Preferences.....	13	Block Outdent.....	38
Downloading Custom Syntax Coloring		Block Comment.....	38
Schemes	15	Block Insert File.....	38
Using Custom Syntax Coloring Schemes	16	Block Write	39
Shortcuts Preferences.....	17	Block Append.....	39
Tools Preferences	18	Block Print	39
System Preferences	20	Bookmark Go to Bookmark.....	40
Keyboard Shortcuts	21	Bookmark Set Bookmark	40
Mouse Actions	23	Extra Spell Check	41
File New	24	Extra Spell Check All	42
File Open	24	Extra Sort Alphabetically.....	42
File Reopen	24	Extra Next Mark	42
File Save	25	Extra Previous Mark.....	42
File Save As.....	25	Extra Compare Files	43
File Save All.....	25	Minimum number of lines for matching block	43
File Print.....	26	Extra Word Count.....	45
Print Preview	26	Convert To Uppercase.....	46
Print Headers.....	27	Convert To Lowercase.....	46
File Mail	28	Convert Initial Caps.....	46
File Mail (advanced)	29	Convert Invert Case	46
File Reload from disk.....	30	Convert To Windows/UNIX/Mac	46
File Rename and Move	30	Convert OEM -> ANSI.....	47
File Delete.....	30	Convert ANSI -> OEM.....	47
File Close (All)	30	Convert Unicode	47
File Exit	30	Convert Wrapping -> Line Breaks.....	47
Selecting Text.....	31	Convert Tabs -> Spaces	48
Edit Undo	31	Convert Spaces -> Tabs	48
Edit Redo	31	Convert ROT-13.....	48
Edit Cut (Append)	32	Convert Scramble & Descramble.....	48
Edit Copy (Append)	32	Options Auto Indent	49
Edit Paste	32	Options Number Lines	49
Edit Swap with Clipboard.....	33	Options Paragraph Symbol.....	49
Edit PasteBook	33	Options Font.....	50
Edit Select All.....	33	Options Export Preferences.....	50
Edit Delete	33	Options Import Preferences	50
Edit Delete Line	33	View Menu	51
Edit Search and Replace.....	34	View Next File	51
Edit Search Forward.....	35	View Previous File	51
Edit Search Backward	35	View Hexadecimal	51
Edit Go to.....	36	View Character Map	51
Edit Insert Date & Time	36	View Browser	52

View New Editor	52	Regular Expressions: \.....	59
Introduction to Regular Expressions	53	Regular Expressions: \b.....	59
History of Regular Expressions	54	Regular Expressions: \w, \s, \d, etc.....	60
Regular Expressions: . (dot)	55	Regular Expressions: Zero-Width Assertions.....	60
Regular Expressions: []	56	Example Regular Expressions.....	61
Regular Expressions: ()	56	Building a Regex to Match a Floating-Point Number	62
Regular Expressions: ? (Zero or One)	57	Remove Trailing Spaces.....	63
Regular Expressions: *	57	Unwrapping Text.....	63
Regular Expressions: +	57	Removing Duplicate Lines	64
Regular Expressions: { }	58	Keeping Only Duplicate Lines.....	64
Regular Expressions: ^.....	58	Removing Comments	65
Regular Expressions: \$	58	EditPad Pro License Agreement.....	66
Regular Expressions: 	58		

Preferences

When you pick Options|Preferences from the menu, the EditPad Pro Preferences screen appears. Since there are a lot of things you can configure, the available settings are split into several tabbed pages. If a certain option seems meaningless to you, simply leave it in its default state. The default settings have been carefully chosen so that EditPad is fully functional even if you do not make any changes in the Preferences screen at all.

Appearance Preferences



On the Appearance tab in the Preferences screen, you can change the looks of EditPad.

You can change many **colors** by selecting an item from the list of color names in the top left. For most items, you can then select the desired (background) color from the “color” drop-down list. For some items, you can also select the text color from the second drop-down list. If you select the color “Default”, the default color for that item will be used. The default color may depend on the colors used by Windows or on other items in the list of EditPad colors. Example: selecting “default” for the “markup tag” color, will use the color specified for “plain text and background”. If you do not like any of the basic colors available from the list, select “Custom” and a screen will appear in which you can select any color you want.

If you only want to change the background and text colors used by EditPad, click on “plain text and background” in the list of colors and then select the background and text colors you want in the two color grids.

In EditPad Pro, you can also configure the colors of the various elements used by the syntax coloring. For these items, you can also change the font style: bold, italic and underline.

These are the colors that you can configure in Appearance Preferences:

Tab - Regular tab

Active Tab - The tab of the file you are currently viewing

Hot Tab - The tab underneath the mouse pointer

Tab - modified - Tab of a modified file

Active Tab - modified - The tab of the file you are currently editing

Hot Tab - modified - The tab of a modified file underneath the mouse pointer

Tab background - The empty space above and to the right of the tabs

Plain text and background - The default text colors. EditPad's background is filled with this color.

Selection text and background - Text and highlight colors of the selected text.

Page break - Color of the horizontal line indicating a page break. (Press Ctrl+Enter to insert one.)

Focus rectangle - Color of the focus rectangle indicating the current byte in hexadecimal mode.

Line numbers - Color of the line numbers.

Right margin - Color of the vertical line indicating the right margin.

All the other items are colors used by built-in as well as custom syntax coloring schemes.

In “**folder to use for open and save dialog boxes**” you can select if the file selection dialog boxes that appear when you want to open or save a file, should start out in the most recently used folder or if they should always start in the same folder, typically your “My Documents” folder. If you select the latter option, you can type in any default folder you like.

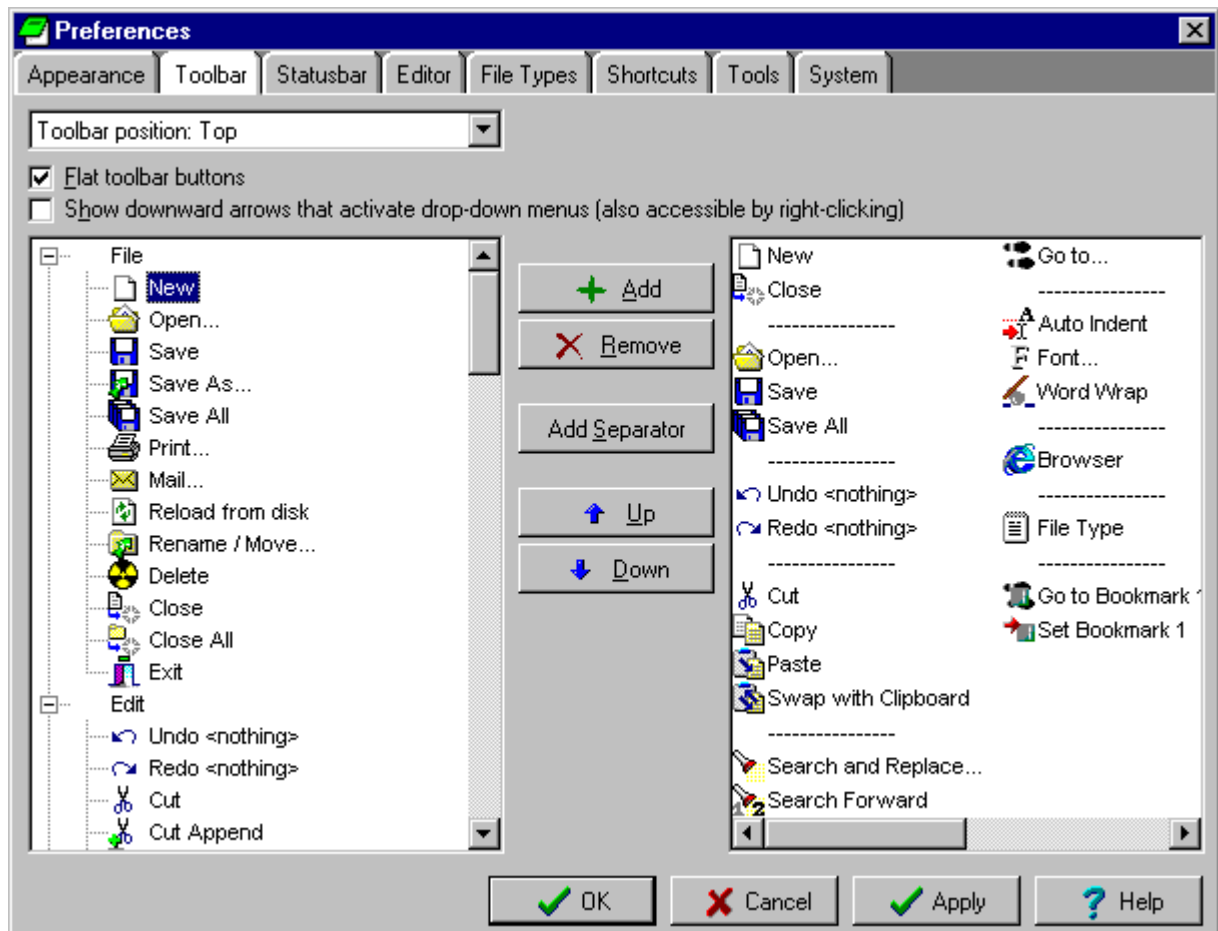
The most recently used folder is the folder the active file is in. If the active file is untitled, the most recently used folder is the folder from which you last opened or in which you last saved a file.

With “**position of the tabs**” you can select if the tabs on EditPad’s main window that you use to switch between files, should appear at the top or at the bottom.

If you turn off “**show images next to menu items**”, no images will be shown in any menu. If you find EditPad’s many colorful icons make it “scream”, you can restore your peace of mind by turning this option off. The images will still be shown on the toolbar buttons and you can still put any item that used to have an image with it on the toolbar.

The tab of the active file has a different color than the other tabs. If this distinction is not clear enough, you can either change the tab colors as explained above or turn on “**active tab is taller than other tabs**”. When this option is selected, the active tab will have a larger height and a piece of background will be visible above all the other tabs. Turning off this option gives the tab control a much more compact look.

Toolbar Preferences



While both EditPad Lite and Pro have a toolbar, it is only configurable in EditPad Pro. Select Tools|Preferences from the menu.

You can place any menu item that has an associated image on the toolbar as a button. Some buttons will automatically receive a downward pointing arrow next to them, which will present an extended list of options related to that command. The buttons with drop-down menus are the most interesting ones to have on your toolbar, so the default toolbar contains all of them.

To **add a button** to the toolbar, click on the item you want to add in the tree view at the left. Then click on the item before which the new button should be placed in the list view at the right. Click in the empty space in the list at the right to deselect all items and the new button will be placed at the end of the list. Then click on the "Add" button to add the button.

To **remove a button**, click on it in the list at the right and then click on the "Delete" button.

To **insert a separator line**, select the insertion point in the right list like when you would add a button and then click the "Add Separator" button.

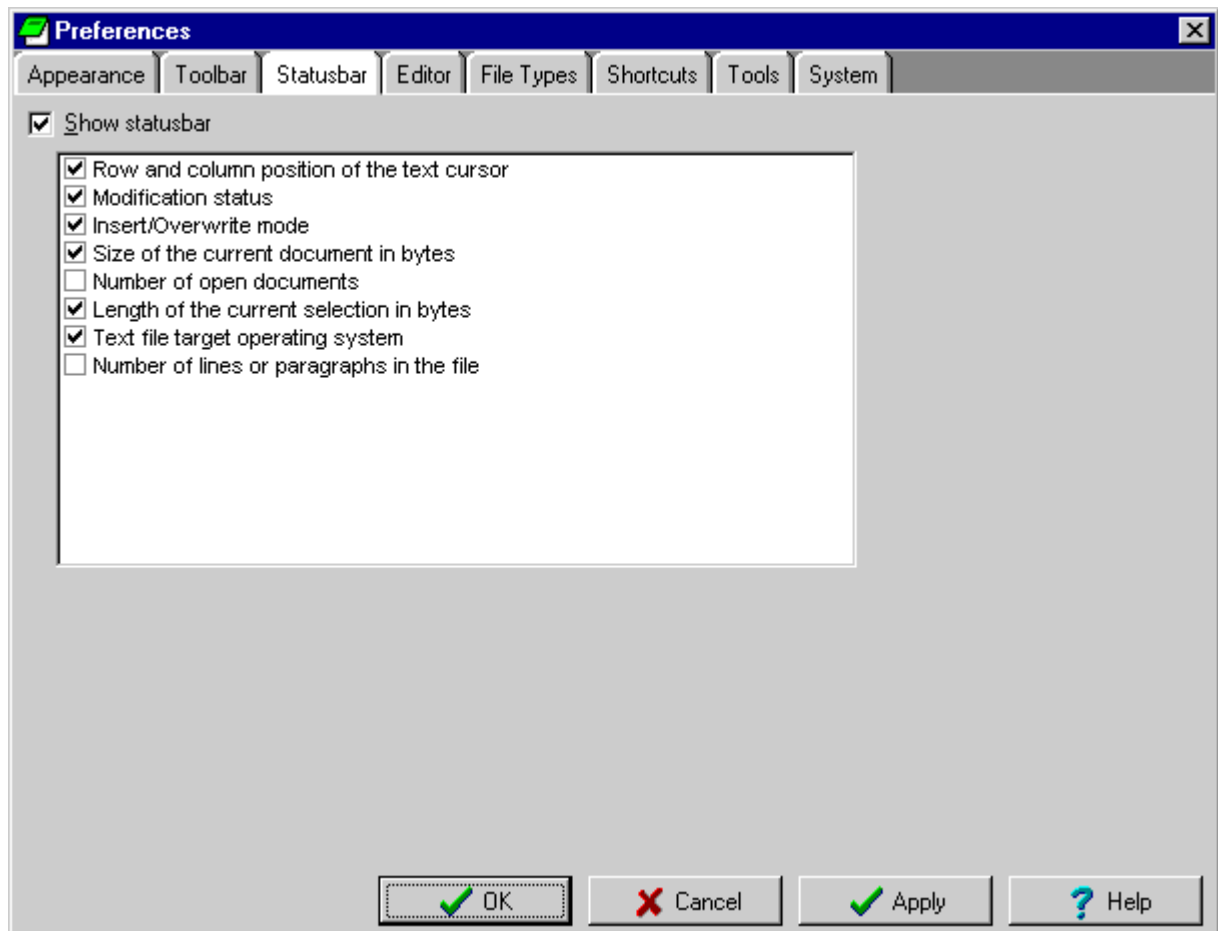
You can **rearrange buttons** on the toolbar by selecting an item in the list at the right and then clicking the "Up" or "Down" button as often as necessary.

From the drop-down list in the upper left corner, you can select on which side of the EditPad Pro window the toolbar should be displayed, or if it should not be displayed at all. If you change the position to the left or right, “show downward arrows” will be turned off. Right-click on a toolbar button to access its drop-down menu.

With “**flat toolbar buttons**” you can choose between the modern flat buttons, or the classic buttons with real edges.

If you want to place many buttons on the toolbar and you do not have a very large screen, you can turn off “**show downward arrows**”. If you do this, the downward arrows will hide, saving much space on the toolbar. You can still access the drop-down menus of the buttons by clicking on them with the right mouse button instead of the left mouse button. (This also works when the downward arrows are visible.) Note that downward arrows can only be shown on horizontal toolbars. If you mark this checkbox when the toolbar is positioned the left or the right, it will be repositioned to the top.

Statusbar Preferences



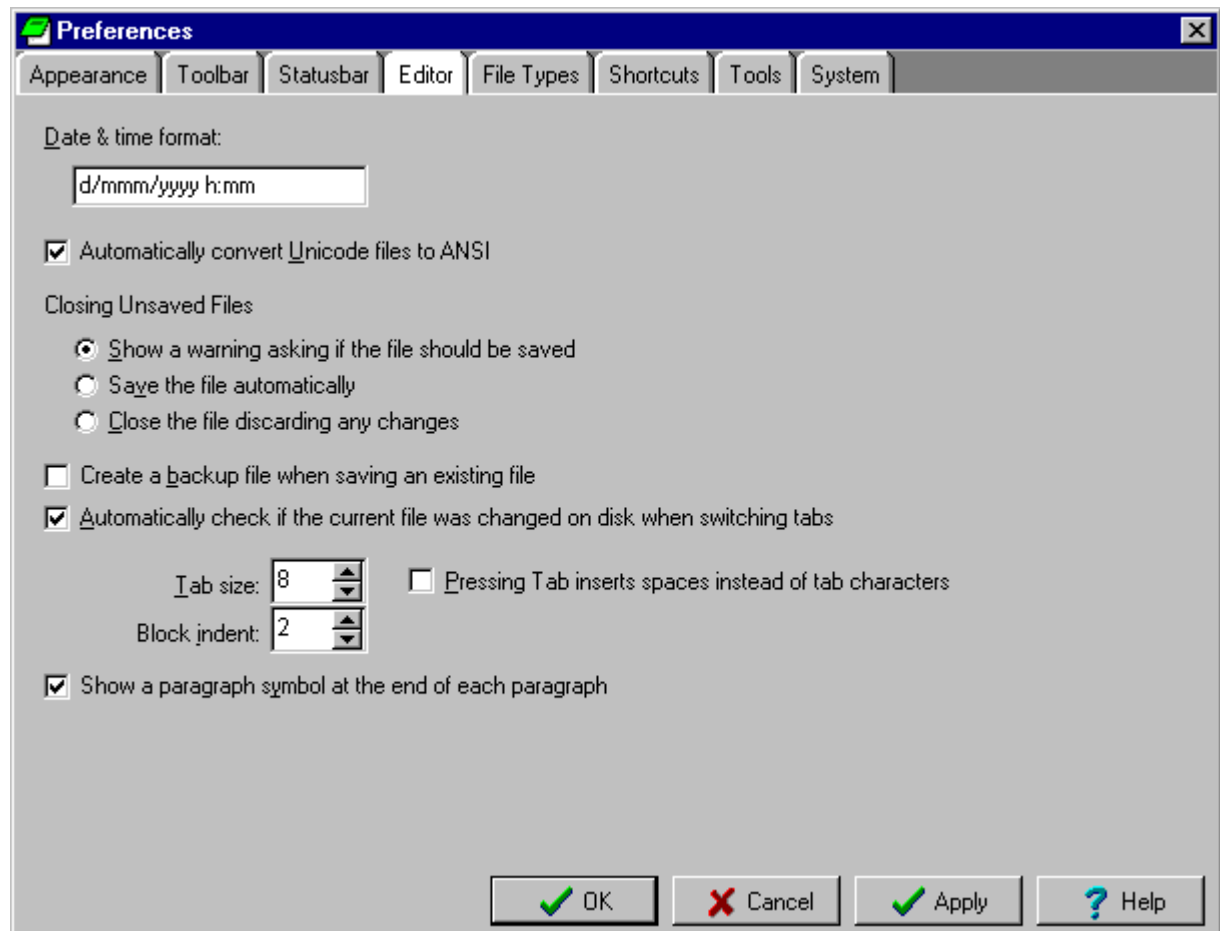
While both EditPad Lite and Pro have a statusbar, it is only configurable in EditPad Pro since it can show much more items on the statusbar and you may not want to clutter the screen with all of them. Select Tools|Preferences from the menu.

You can show or hide items on the statusbar by marking or clearing the checkboxes at the left of the items in the list.

You can change the order in which the items are shown on the statusbar by dragging and dropping them in the list. Click on an item, hold the mouse button down and drag it to the spot where you want it and release the mouse button. The topmost item in the list will become the leftmost item on the statusbar.

You can also hide the statusbar altogether by turning off "show statusbar".

Editor Preferences



On the Editor tab of the Preferences you can set the options that affect basic editing tasks that are not file type specific.

In “**date and time format**” you can specify the format that should be used by the Edit|Insert Date and Time function. You will need to use special character sequences that function as placeholders for the actual date and time values.

If the date and time format is left empty, Edit|Insert Date and Time will enter the date in short format followed by the time in long format as specified in the regional settings in the Windows Control Panel.

EditPad cannot work with Unicode files directly, but it can convert a Unicode file on-the-fly to ANSI while opening it so you can edit it in EditPad. UCS-2, UCS-2 BE and UTF-8 encodings are supported. If you do not understand this, leave the option to “**automatically convert Unicode files to ANSI**” on. Manual conversion is possible through the Convert Menu.

The “**closing unsaved files**” option gives you three choices about what EditPad should do when you instruct it to close a file to which you have made changes that were not yet saved to disk:

- Show a warning message and give you one last opportunity to decide if the file should be saved or not. This is the default since most other software behaves this way too.
- EditPad can automatically save the file without asking you first. This is the best way to make sure you will not lose any work.
- The last option will make EditPad close the file without warning. Any unsaved work will be lost, so use with caution.

The option to **create a backup file when saving an existing file** will create a copy of the original file with a .bak extension if you overwrite a file while saving another one.

When you switch tabs or switch back to EditPad after using another application, EditPad will automatically check if the active file has not been modified on disk by another application. If it is, and you have not edited it in EditPad, it will be automatically reloaded from disk. If you did edit it, you will be asked if you want to reload it or not. This may cause some trouble if you are editing files on a slow medium like a floppy disk or over a slow network link. There will be a noticeable delay each time you switch tabs. In that case, turn off the option to **automatically check if the current file was changed on disk**.

“Tab size” is the size of a tab in spaces. You can also choose if pressing Tab on the keyboard should insert an actual tab character or the equivalent number of spaces.

“Block indent” is the number of spaces that the Block|Indent and Block|Outdent will use to indent or outdent the current selection.

Turn on **“show paragraph symbol at the end of each paragraph”** if you always want to see a ¶ character at the end of each paragraph, indicating the invisible line break character(s). Note that if line break characters are selected, the paragraph symbol will always appear to show you that they have indeed been selected. You can quickly change this option by picking Options|Paragraph Symbol from the menu.

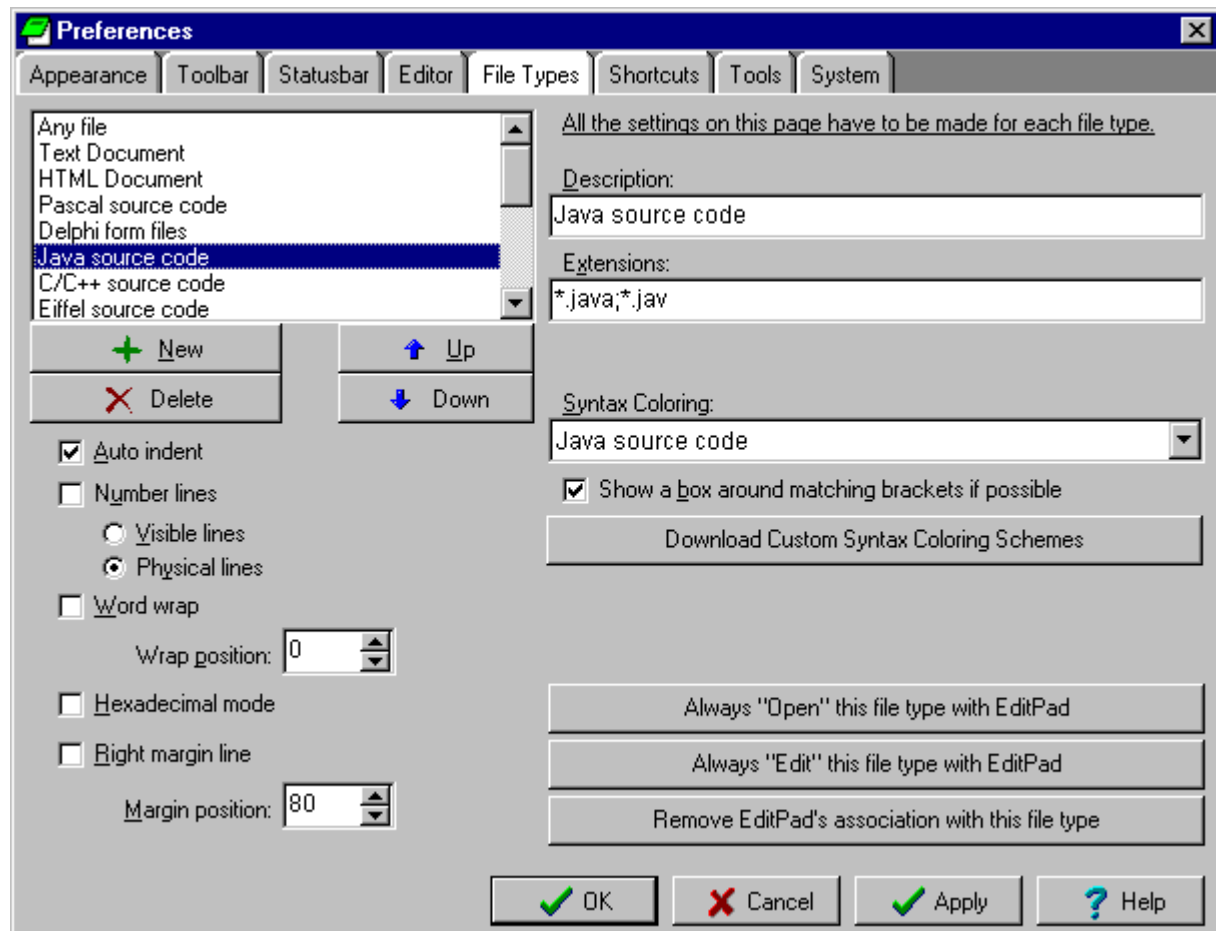
Date and Time Specifiers

These are the date and time placeholders that you can use in the Date and Time Format setting in the Editor tab of the Preferences window.

Format specifiers may be written in upper case as well as in lower case letters. Both produce the same result.

Specifier	Displays
c	Displays the date in short format followed by the time in long format as specified in the regional settings in the Windows Control Panel.
d	Displays the day as a number without a leading zero (1-31).
dd	Displays the day as a number with a leading zero (01-31).
ddd	Displays the day as an abbreviation (Sun-Sat) in the language of the current Windows locale.
dddd	Displays the day as a full name (Sunday-Saturday) in the language of the current Windows locale.
dddddd	Displays the date using the short date format specified in the regional settings in the Windows Control Panel.
dddddd	Displays the date using the long date format specified in the regional settings in the Windows Control Panel.
m	Displays the month as a number without a leading zero (1-12). If the m specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mm	Displays the month as a number with a leading zero (01-12). If the mm specifier immediately follows an h or hh specifier, the minute rather than the month is displayed.
mmm	Displays the month as an abbreviation (Jan-Dec) in the language of the current Windows locale.
mmmm	Displays the month as a full name (January-December) in the language of the current Windows locale.
yy	Displays the year as a two-digit number (00-99).
yyyy	Displays the year as a four-digit number (0000-9999).
h	Displays the hour without a leading zero (0-23).
hh	Displays the hour with a leading zero (00-23).
n	Displays the minute without a leading zero (0-59).
nn	Displays the minute with a leading zero (00-59).
s	Displays the second without a leading zero (0-59).
ss	Displays the second with a leading zero (00-59).
z	Displays the millisecond without a leading zero (0-999).
zzz	Displays the millisecond with a leading zero (000-999).
t	Displays the time using the short time format specified in the regional settings in the Windows Control Panel.
tt	Displays the time using the long time format specified in the regional settings in the Windows Control Panel.
am/pm	Uses the 12-hour clock for the preceding h or hh specifier, and displays 'am' for any hour before noon, and 'pm' for any hour after noon. The am/pm specifier can use lower, upper, or mixed case, and the result is displayed accordingly.
a/p	Uses the 12-hour clock for the preceding h or hh specifier, and displays 'a' for any hour before noon, and 'p' for any hour after noon. The a/p specifier can use lower, upper, or mixed case, and the result is displayed accordingly.
/	Displays the date separator character specified in the regional settings in the Windows Control Panel.
:	Displays the time separator character specified in the regional settings in the Windows Control Panel.
'xx'/'"xx"	Characters enclosed in single or double quotes are displayed as-is, and do not affect formatting.

File Types Preferences



One of the most powerful aspects about the way EditPad Lite and Pro work with preferences, is that many preferences can be made for each specific file type that you regularly edit or view with EditPad. For example: you may want to use word wrapping at the window border for plain text files, but no word wrapping for source code. Making the correct settings here once, will prevent you from having to change them each time you open a file.

When you open or save a file, EditPad will look up its extension in the list of known file types and will apply the settings made for that file type, or the settings for "Any file" if the extension is not related to a file type known to EditPad.

The list of file types is also used to build the "filters" (the drop-down list at the bottom) for the file selection windows used for opening and saving files.

The list in the upper left shows the file types currently known to EditPad. If you want to change the settings for a certain file type, click on it in the list.

The "**description**" item is used whenever you need to make a choice between file types, like the "filter" in open and save windows or the Options|File Type command.

You should take care when specifying the **extensions** for the file type. In the Microsoft Windows operating system, the type of a file is determined by its extension. The extension is the last dot (.) in the filename plus any characters that follow it (usually three letters and/or digits).

In the “extensions” field, you need to specify the extension preceded by an asterisk (*), e.g.: *.txt
If more than one extension is used for this file type, separate them with a semicolon (;), e.g.: *.html;*.htm;*.shtml
The first extension in the list will be the default extension used by EditPad, so it should be the one you use most often for this file type.

Next you can select which coloring scheme to use from the “**syntax coloring**” drop-down list. EditPad Lite has only one scheme that makes URLs clickable so you can quickly open them in your web browser. EditPad Pro comes with many schemes for many popular file types. If no coloring scheme is available for the file type you are defining, click the “**download custom syntax coloring schemes**” button. EditPad Pro will then connect to the Internet and allow you to download many custom syntax color schemes created and uploaded by other EditPad Pro users. See the next section for more information.

To create your own syntax coloring schemes, use the JGsoft Custom Syntax Coloring Schemes Editor. You will find the information to download it in the email with the license information that you received shortly after you purchased EditPad Pro. If you cannot find it, you can have it resent immediately by typing in your email address at <http://www.editpadpro.com/download.html>

Most syntax coloring schemes can “**show a box around matching brackets**”. If this box bothers you, turn it off here. Note that for some schemes, such as “clickable URLs”, there will be no difference whether this option is on or off, as these schemes are not capable of marking matching brackets.

Turn on “**auto indent**” if you want the next line to automatically start at the same column position as the previous line whenever you press Enter on the keyboard while in the editor. EditPad will accomplish this by counting the number of spaces and tabs at the beginning of the previous line and inserting them at the beginning of the new line you created by pressing Enter. This is most useful for writing source code.

Turn on “**number lines**” to have EditPad display a number before each line in the left margin. In that case, you need to indicate if you want **visible lines** or **physical lines** to be numbered. This only makes a difference when word wrapping is on. When numbering visible lines, each line is counted. When numbering physical lines, the lines are numbered as if word wrapping would be off.

You can choose if “**word wrap**” should be on by default for this file type. The “**wrap position**” is the maximum number of characters you want to appear on a single line. If you set this to zero, wrapping will occur at the right edge of the EditPad window. Note that word wrapping only affects the display of the file in EditPad. Word wrapping is not saved into the file.

Activate “**hexadecimal mode**” if the file type deals with binary files. You can still toggle hexadecimal mode though the View menu after opening a file of this type, regardless of this setting.

If you want to see a vertical line at a certain column position, turn on “**right margin line**” as specify the column number in “**margin position**”. The column position of the line will only be exact if you are using a fixed-width font like *Courier New*.

Using the three buttons in the lower right of this page, you can quickly create or remove file associations to the current file type. The associations will be made with or removed from each of the extensions for this file type.

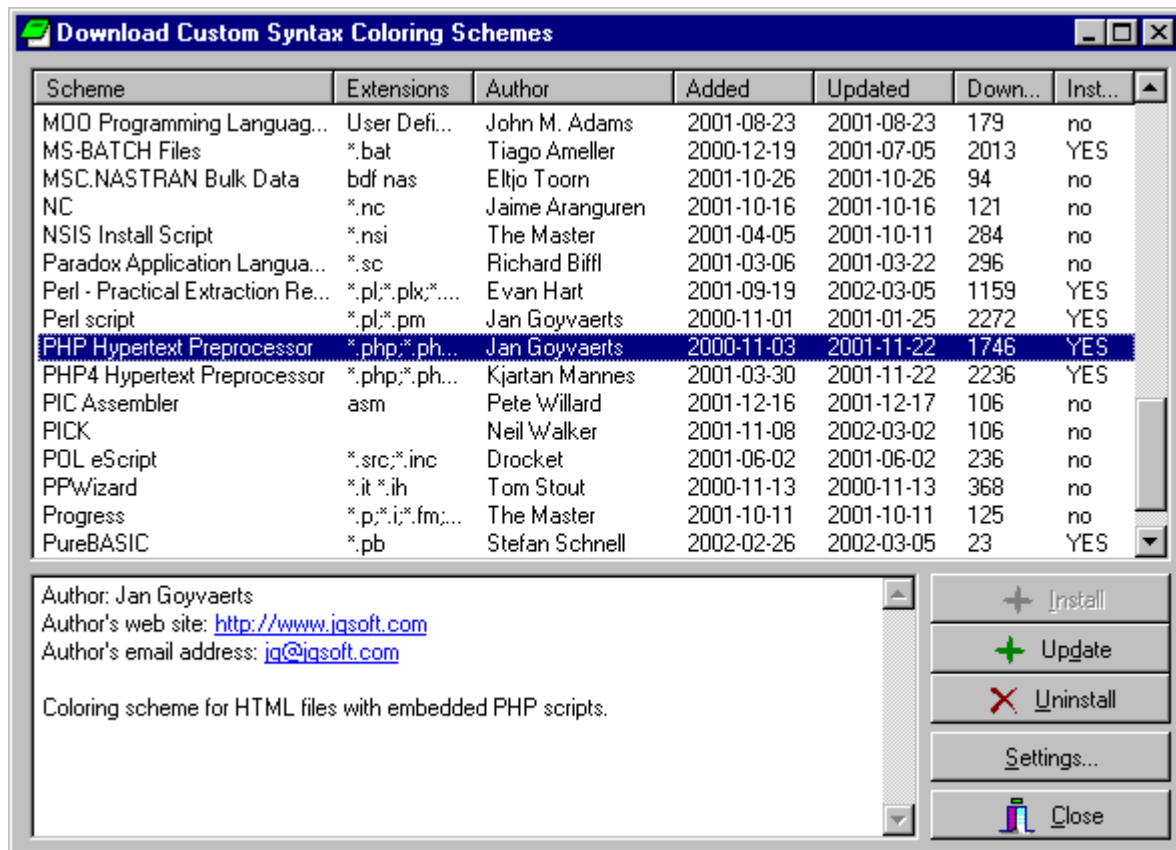
If you click the button “**Always “Open” this file type with EditPad**”, EditPad will associate itself with the “open” action. This means that whenever you double-click on a file of this type in Windows Explorer, it will be opened in EditPad.

If you click the button “**Always “Edit” this file type with EditPad**”, EditPad will associate itself with the “edit” action. This means that you can then right-click on files of this type in Windows Explorer and select “Edit” from the context menu to open the file in EditPad. If you do this for HTML files, you will be able to view the source for any web page by clicking on the Edit button in the toolbar of Microsoft Internet Explorer.

Click the button “**Remove EditPad’s association with this file type**” to remove the associations made by clicking on one or both of the two previous buttons. Associations made in another way will not be removed. Associations made by the installation program will be removed when you uninstall EditPad via the Control Panel (don’t do that!).

Downloading Custom Syntax Coloring Schemes

When you click on the Download Custom Syntax Coloring Schemes button in File Types Preferences, EditPad Pro will connect to the Internet. It will download a list of custom syntax coloring schemes that have been created and generously shared by other EditPad Pro users. This list is then displayed in the window shown below.



The list presents you the following information about the available schemes:

- **Scheme:** The name of the scheme. It is possible that more than one scheme with the same name is available. If so, you can install all of them or only one, as you prefer.
- **Extensions:** The extensions typically used for the files that the scheme provides syntax coloring for. They are only shown for your information. You can apply any scheme to any file type, as you see fit.
- **Author:** The author of the scheme. If you have comments or suggestions about a particular coloring scheme, you should contact this person.
- **Added:** The date the scheme was first added to the list of available schemes.
- **Updated:** The date the scheme was last updated.
- **Downloads:** Number of times this scheme has been downloaded.
- **Installed:** Indicates whether this scheme is installed on your computer or not.

If you click the **Install** button, EditPad Pro will immediately download the scheme and save it into the same folder as its executable file EditPadPro.exe. Click the **Update** button to download a scheme that you already installed again. If you click the **Uninstall** button, EditPad Pro will delete the scheme from your computer when you close the window for downloading custom syntax coloring schemes.

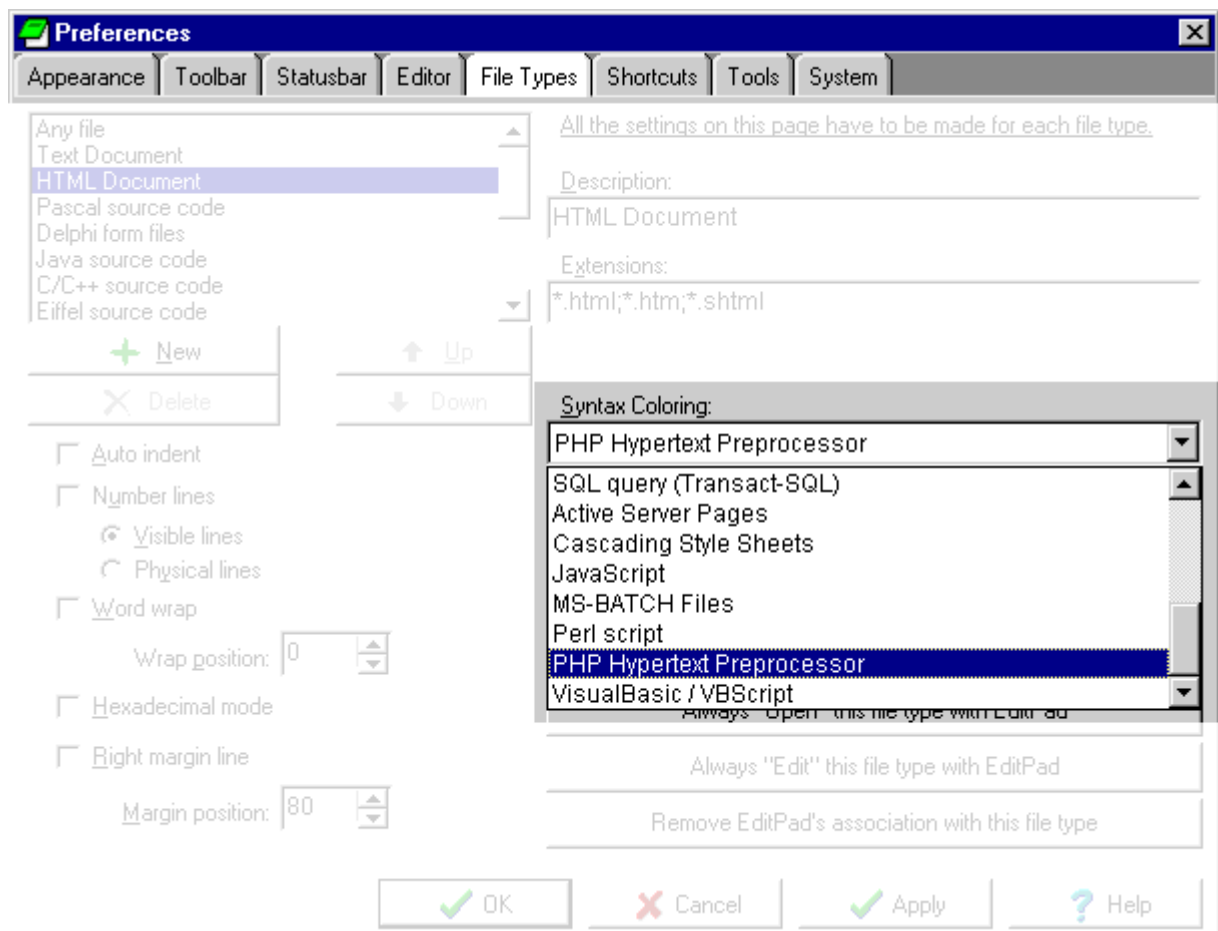
If you are behind a proxy server, and EditPad Pro is unable to detect your proxy settings, you can change them by clicking the **Settings** button.

Using Custom Syntax Coloring Schemes

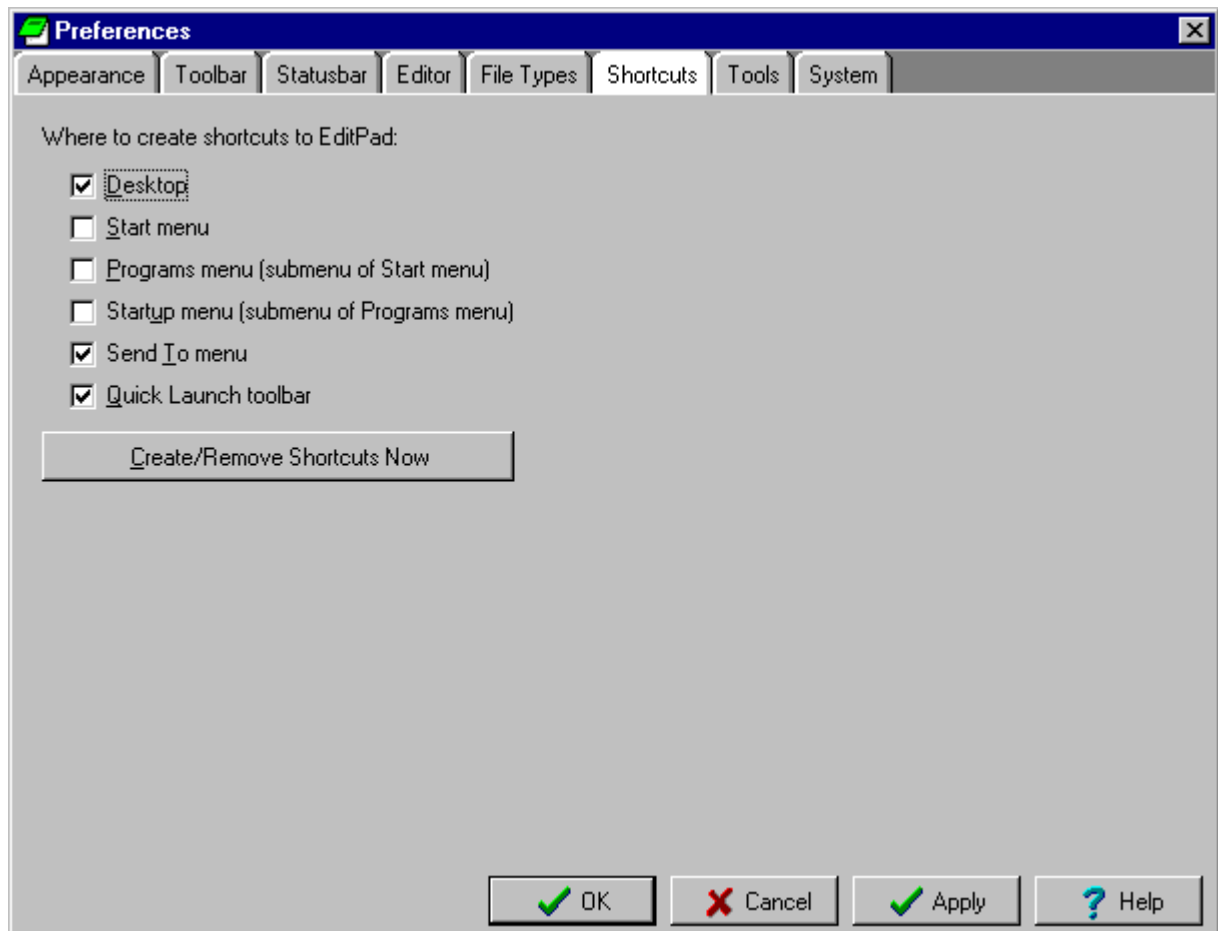
After downloading a custom syntax coloring scheme, it is *not* automatically put to use.

All the schemes that you downloaded, will be listed in the Syntax Coloring drop-down list in File Types Preferences. They will appear near the bottom of the list, below the built-in schemes. This drop-down list is shown in the screen shot below.

To use one of the schemes, create or edit a file type, and select the coloring scheme you want from the Syntax Coloring drop-down list.



Shortcuts Preferences

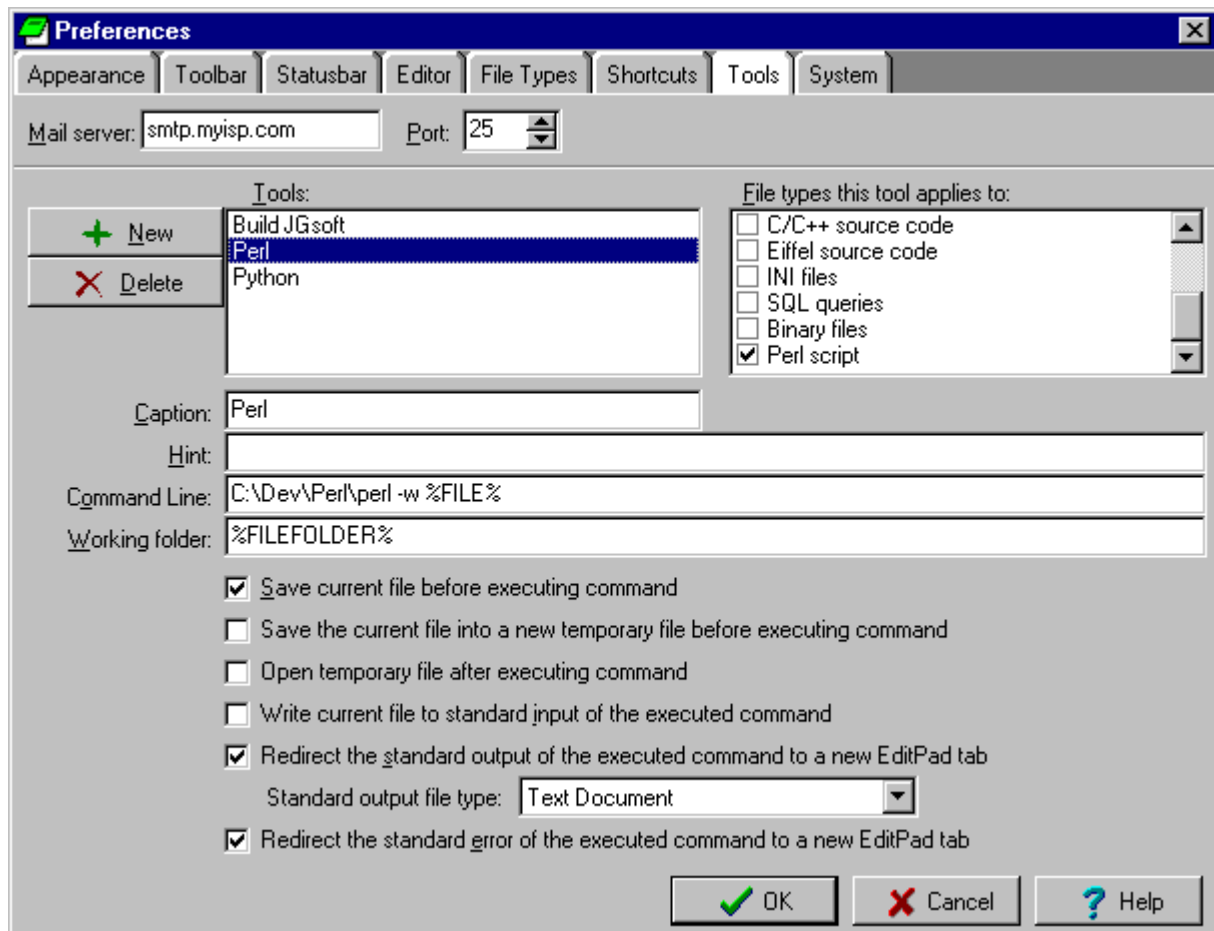


This page in the Preferences window makes it easy for you to create shortcut icons to EditPad. Select the various places where you want a shortcut icon and click the **“Create/Remove Shortcuts Now”** button. To remove a shortcut, clear the appropriate checkboxes and press the same button.

If you place a shortcut in the **“Startup menu (submenu of Programs menu)”**, the green EditPad icon will automatically appear next to the system clock whenever you turn on your computer.

Put a shortcut in the **“Send To menu”** so you can quickly open any file in EditPad from within Windows Explorer by right-clicking on the file, selecting Sent To from the context menu and then selecting EditPad. The **“Quick Launch Toolbar”** is not available in Windows 95 and NT4.

Tools Preferences



If you want to use the File|Mail feature, you need to specify the outgoing **mail server** (SMTP) on the Tools page in Preferences.

If you do not know which mail server you should use, contact your Internet Service Provider (ISP).

EditPad Pro has the capability to run any external command or tool. The tools are listed in the list at the left. If you have not yet defined any tools, the list will be empty. Click the **“New”** button to add a tool or click on an existing tool in the list that you want to change.

Since most tools will only make sense when used on a file of a certain type, you can select the **file types this tool applies to** in the list at the right. Do this by marking the appropriate checkboxes. Not marking any checkbox has the same effect as marking all of them.

EditPad Pro will add to the Tools menu all the tools that apply to the file type that is the file you are currently editing.

“Caption” is the caption that this tool's menu item in the Tools menu will have. If you want to make a certain character a hotkey in the menu, precede it with an ampersand (&). The character that follows the ampersand will be shown underlined in the menu (unless you are using Windows 2000 and disabled the underlining of hotkeys).

“Hint” is the text that will be shown in the status bar when the mouse is pointing that this tool's menu item in the Tools menu. You may leave this blank if the caption is descriptive enough.

For the **“Command Line”**, you need to type in what you would type in if you would launch the tool from the Run item in the Windows Start menu. It is best to include the full path to the executable and any file parameters. Note that you must not use <, > or | characters for file redirection. If you need file redirection, mark the appropriate checkboxes below.

“Working Folder” is the folder that will appear to be the current one to the tool when it is run by EditPad.

Several placeholders are available in the “Command Line” and “Working Folder” settings:

%FILE% is the complete path (folder + file name) to the file that is active in EditPad at the time the tool is run.

%FILEFOLDER% is the folder which that file is in, including the trailing backslash.

%FILENAME% is the file name of the file, including the extension, without the folder name.

%FILENAMENOEXT% is the file name of the file without the extension, without the folder name.

%TEMPFILE%, %TEMPFILEFOLDER% and %TEMPFILENAME% are the same but for the temporary file to which EditPad Pro can save the file before running the tool, and which EditPad Pro can open after running the tool.

In the case of %FILE% and %TEMPFILE%, EditPad Pro will automatically put double quotes around the file path if it contains one or more spaces. The other placeholders are always substituted without using quotes, even if the folder or filename contains spaces.

Mark **“Save current file before executing command”** if EditPad should automatically do a File|Save before the tool is run.

Mark **“Save the current file into a new temporary file before executing command”** if EditPad should automatically do a File|Save As and save the file under a new, temporary name before the tool is run. This is useful if the tool will modify the file's contents but you do not want to lose the original, or if you want to test changes you made to a file with a tool without having to save those changes first. The temporary file will be deleted after the tool has finished running.

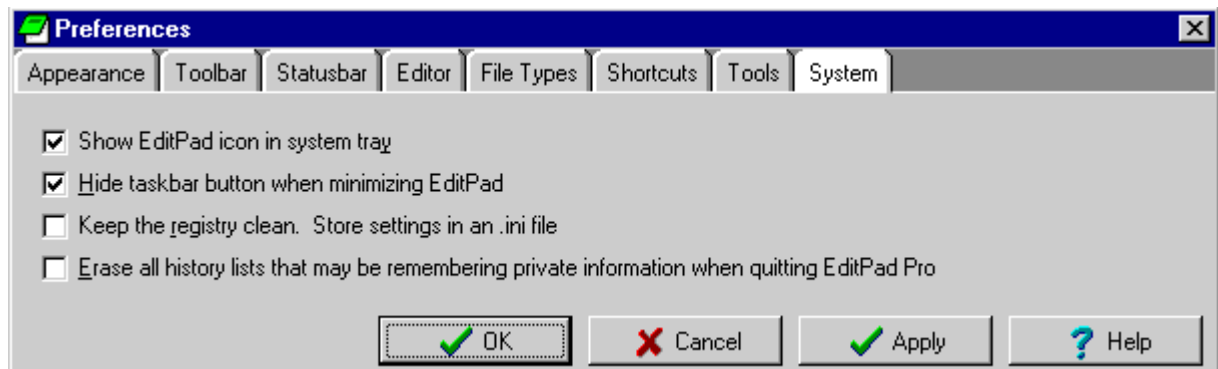
Mark **“Open temporary file after executing command”** if the tool will modify the temporary file and you want to see the changes in EditPad. If this option is checked, EditPad will open the file into a new tab and then delete the temporary file from disk. If you want to save the tool's result, use File|Save As.

If you don't know what **“Write current file to standard input of the executed command”** means, leave this option off.

If the tool runs in a console (text-only) window and writes its output into it, you can make EditPad capture the output into a new tab if you turn on **“Redirect the standard output of the executed command to a new EditPad tab”**. If you do so, you can also select which file type should be used to determine the settings for the new tab. When standard output is redirected, the tool's console window will still appear, but it will remain empty. If you switch to EditPad while the tool is running, you will see the output flashing by. If you press Page Up on the keyboard while this happens, the output will stop scrolling so you can read it even while it is not yet complete. You can also continue to work with other files in EditPad if the tool takes a long time to run. EditPad will continue to capture the output in the background.

Just like with the standard output, you can also redirect the **standard error**.

System Preferences



If “**Show EditPad icon in the system tray**” is activated, a green EditPad icon will be visible next to the system clock (this area is called the “system tray”) when EditPad is running. When you close EditPad in this situation, it will not really close but only hide itself so the system tray icon remains visible. You can then make EditPad quickly reappear by clicking on the system tray icon. You can right-click on it to quickly start with a new file or open an existing one. To completely shut down EditPad, select File|Exit from the menu.

Turn off this option to make EditPad behave like a regular Windows program.

When the “Show EditPad icon in the system tray” option is activated, you can choose if you want EditPad’s button on the taskbar to hide or not when you minimize EditPad by marking or clearing “**Hide taskbar button when minimizing EditPad**”. Note that the taskbar button will always hide when you close EditPad.

You can have EditPad **keep the registry clean and store its settings in an .ini file**. This is most useful if you are carrying EditPad with you on a floppy disk and want to run it on other people’s computers. If you activate this option, EditPad will remove all its settings from the registry and store them in an .ini file in the same folder as EditPad’s .exe file. The next time you run EditPad, it will detect the .ini file and continue to use that instead of the registry.

The “**erase all history lists that may be remembering private information when quitting EditPad Pro**” can be useful if you share your computer with other people and you don’t want them to figure out which files you have been working on in EditPad Pro. Every time you close EditPad Pro using File|Exit when this option is active, EditPad Pro will clear history lists such as File|Reopen, Project|Reopen and the search and replace history lists (which you can access by right-clicking in the search or edit boxes). If these lists were previously saved to the registry or to the .ini file, they will be deleted.

Keyboard Shortcuts

The shortcuts in this list are the shortcuts that are recognized by every full text editor control in EditPad, such as the main editor, the search box, the replace box, the PasteBook editor, etc.

Some of these shortcuts are also used by menu items. If you select the command from the menu, it will always work on the main editor. If you select the command by pressing the shortcut key combination, it will work on the editor that has input focus. This is the editor that shows the blinking text cursor.

When a key combination only works in text mode, this is indicated by [text]. If it only works in hexadecimal mode, the indication is [hex].

Cursor movement keys

Key combination	Action
Arrow key	Moves the text cursor (blinking vertical bar).
Ctrl+Left arrow [text]	Moves the text cursor to the start of the previous word or the end of the previous line, whichever is closer.
Ctrl+Right arrow [text]	Moves the text cursor to the start of the next line or the end of the current line, whichever is closer.
Ctrl+Up/Down arrow	Scrolls the text one line into the opposite direction, keeping the text cursor in place.
Page up/down	Moves the text cursor up or down an entire screen.
Ctrl+Page up/down	Scrolls the text an entire screen into the opposite direction, keeping the text cursor in place.
Home	Moves the text cursor to the beginning of the line.
Ctrl+Home	Moves the text cursor to the start of the text.
End	Moves the text cursor to the end of the line.
Ctrl+End	Moves the text cursor to the end of the text.
Shift+Movement key	Moves the text cursor and expand or shrink the selection towards the new text cursor position. If there was no selection, one is started. Pressing Ctrl as well, will move the text cursor correspondingly.
Alt+Shift+Mov. key [text]	The same as when Alt is not pressed, except that the selection will be rectangular instead of flowing along with the text.

Editing commands

Key combination	Action
Enter	In the main editor: inserts a line break. In the search or replace box: search for the next occurrence.
Shift+Enter	Inserts a line break.
Ctrl+Enter	Inserts a page break.
Delete	Deletes the current selection if there is one and selections are not persistent. Otherwise, the character to the right of the caret is deleted.
Ctrl+Delete	Deletes the current selection if there is one. Otherwise, the part of the current word to the right of the text cursor is deleted [text].
Shift+Ctrl+Delete	Deletes the current selection if there is one and selections are not persistent. Otherwise, all the text on the current line to the right of the text cursor is deleted [text].

Backspace	Deletes the current selection if there is one and selections are not persistent. Otherwise, the character to the left of the caret is deleted.
Ctrl+Backspace	Deletes the current selection if there is one and selections are not persistent. Otherwise, the part of the current word to the left of the text cursor is deleted [text].
Shift+Ctrl+Backspace	Deletes the current selection if there is one and selections are not persistent. Otherwise, all the text on the current line to the left of the text cursor is deleted [text].
Alt+Backspace	Undoes the last action.
Alt+Shift+Backspace	Redoes the last undone action.
Ctrl+Z	Undoes the last action.
Ctrl+R	Redoes the last action.
Insert	Toggles between insert and overwrite mode.
Tab	If there is a selection, the entire selection is indented. Otherwise, a tab is inserted.
Shift+Tab	If there is a selection, the entire selection is unindented (outdented). Otherwise, if there is a tab, or a series of spaces the size of a tab, to the left of the text cursor, that tab or spaces are deleted.
Ctrl+A	Selects the entire text.
Ctrl+Y	Deletes the current line.

Clipboard commands

Key combination	Action
Ctrl+X	Edit Cut
Shift+Ctrl+X	Edit Cut Append
Ctrl+C	Edit Copy
Shift+Ctrl+C	Edit Copy Append
Ctrl+V	Edit Paste
Shift+Delete	Edit Cut
Ctrl+Insert	Edit Copy
Shift+Insert	Edit Paste

Mouse Actions

Below you can find a list of action you can carry out in the editor using the mouse.

Dragging means to move the mouse before releasing the mouse button you pressed. If you move the mouse pointer to the edge of the editor space while dragging, the text will start to scroll automatically.

Modifier keys like shift or control must be pressed before pressing the mouse button and kept depressed until the mouse button is released.

Mouse Action	Result
Left click	Moves the text cursor to the spot where you clicked.
Shift+Left click	Moves the text cursor and expands or shrinks the selection. If there is no selection, the text between the old and new cursor positions becomes selected. If you click outside of the selection, the selection plus the text between the selection and the new cursor position becomes selected. If you click inside the selection, the new selection is the text between the original start of the selection and the new cursor position.
Left click+drag	When clicking outside the selection, a new selection is created from the point where you press the mouse button until the point where you release it. When clicking inside the selection, the selected text deleted and inserted again at the spot (outside the selection) where you release the mouse button.
Shift+Left click+drag	Expands or shrinks the selection like Shift+Left click, but then the text cursor is moved and the selection adjusted until you release the mouse button.

If you press Alt while changing the selection with the mouse, the selection becomes rectangular.

Rotate wheel	Scrolls the text a single line up or down. The text cursor keeps its position relative to the text, so it scrolls along and may scroll out of view.
Shift+Wheel	Moves the text cursor a line up or down, like pressing the up or down arrow keys on the keyboard.
Ctrl+Wheel	Scrolls the text an entire screen up or down.
Shift+Ctrl+Wheel	Moves the text cursor a screen up or down, like pressing page up or down on the keyboard.

File|New

Creates a new tab with a blank, untitled file.

The file type for the new file is the second file type defined on the File Types page in Preferences. By default, this file type is “Text Document” which is used for .txt files.

When you use File|Save As and give the file a name with an extension other than .txt, the file’s file type will be changed to the file type associated with that extension. The file will then also use the settings defined for that file type.

In EditPad Pro, if you put the File|New button on the toolbar (it is on the toolbar by default), it will receive a drop-down menu listing all available file types. If you select an item from that menu, a new tab will be created with a blank, untitled file which uses the settings for the file type you just selected, instead of the “Text Document” file type which is used if you pick File|New from the menu or left-click the New button on the toolbar.

File|Open

The File|Open command will show an open file common dialog box and allows you to open one or more files from the folder of your choice. To select more than one file, hold down the Shift or Control key on the keyboard while you click with the mouse.

The initial folder of the open dialog (the folder of which the contents are listed after you select File|Open) is determined by the “folder to use for open and save dialog boxes” setting on the Appearance Preferences. There you can select if the initial folder should be the most recently used folder or if it should always be the same folder, typically your “My Documents” folder. If you select the latter option, you can type in any default folder you like.

The most recently used folder is the folder the active file is in. If the active file is untitled, the most recently used folder is the folder from which you last opened or in which you last saved a file.

If the active file is an empty one, it will be replaced by the one you open. This ensures EditPad does not get cluttered with empty tabs.

File|Reopen

A list of up to 16 recently opened files is available in the Reopen submenu of the File menu. If you click on one of these items, that file will be opened. The Reopen menu is also available as the drop-down menu of the File|Open button on the toolbar.

At the bottom of the menu, you will see the “Remove Obsolete” and “Remove All” items. The former will remove all files from the reopen menu that no longer exist, the latter will clear the entire reopen menu.

File|Save

Saves the active file to disk, overwriting the original, if any, without warning.

In Editor Preferences you can specify that a backup copy of the original file should be created. This backup copy will have the extension .bak

If the active file is untitled, File|Save will invoke File|Save As.

File|Save As

Saves the active file under a new name. If the active file has previously been saved, the original copy will remain available. If you later use File|Save after using File|Save As, it will save under the new name as well. This is indicated by the filename on the tab which changes after you use File|Save As.

When you select File|Save As, a save file common dialog box will appear. You can select the folder in which you want to save and type in a filename.

If you do not specify an extension for the filename, EditPad will add one depending on the selection you made in the filters (the bottommost drop-down list) in the Save As dialog box. If the active filter is “any file” (*.*), no extension will be added. If any other filter is active, the default extension for that file type will be added to the filename. The default extension is the first extension listed in the Extensions setting of the file type’s settings.

If you do specify an extension, EditPad will save the file with that extension. If the extension is part of another file type, all the file type dependent settings such as auto indent, number lines and word wrap will be changed to match the settings for that new file type as made in File Types Preferences.

The initial folder of the save as dialog (the folder of which the contents are listed after you select File|Save As) is determined by the “folder to use for open and save dialog boxes” setting on the Appearance Preferences. There you can select if the initial folder should be the most recently used folder or if it should always be the same folder, typically your “My Documents” folder. If you select the latter option, you can type in any default folder you like.

The most recently used folder is the folder the active file is in. If the active file is untitled, the most recently used folder is the folder from which you last opened or in which you last saved a file.

File|Save All

Rapidly performs a File|Save for each open file.

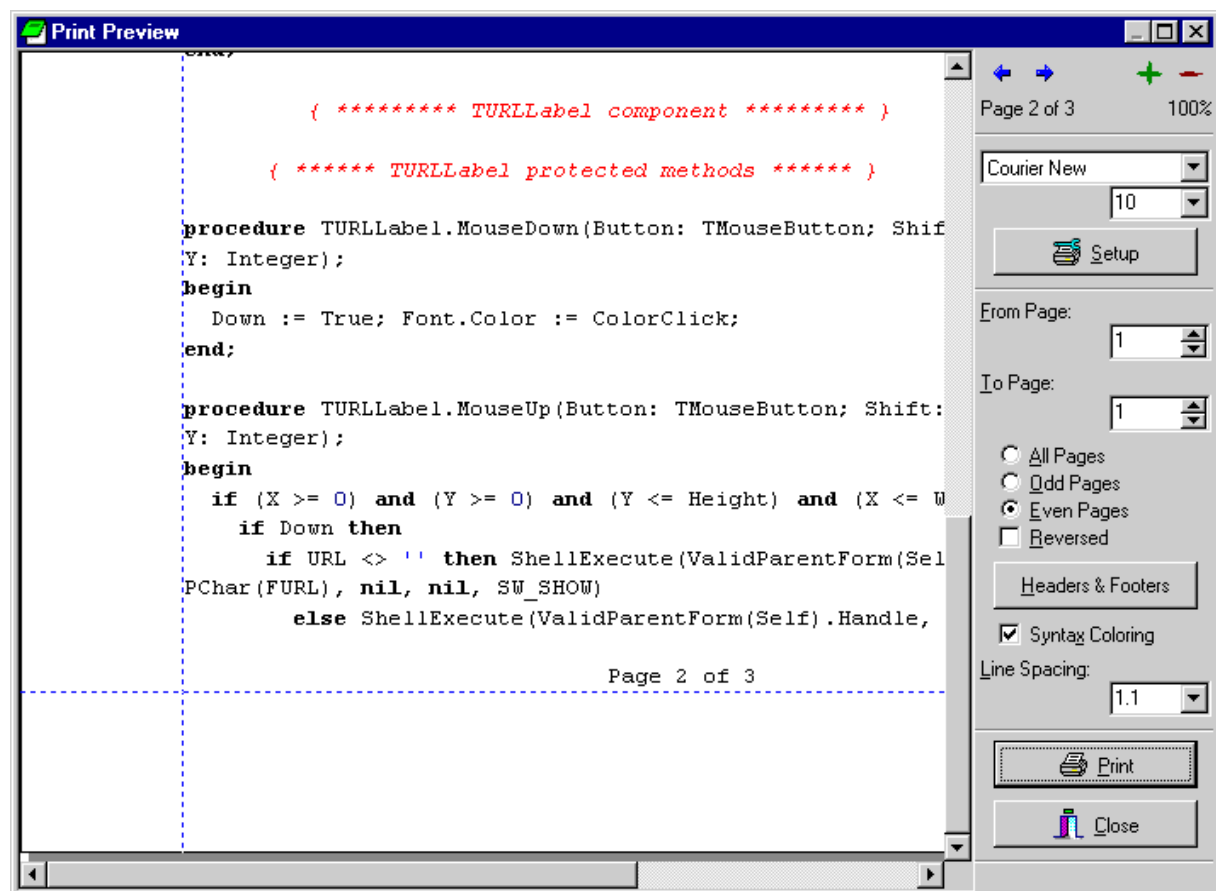
For untitled files, File|Save As is invoked. If you click on Cancel in the save as dialog box, the save all function will not save any further files either.

File|Print

Prints the entire active file. A preview of the printout will be shown first. If you are only interested in a specific part of the file, select that part first and then pick Block|Print from the menu.

Print Preview

This window is shown when you select File|Print or Block|Print.



With the blue arrow buttons at the top, you can **browse** through the pages that will be printed. You can also press the Page Up and Page Down keys on the keyboard.

The green plus and red minus buttons **zoom** the preview in and out respectively. You can also press the plus and minus keys on the numeric keypad on your keyboard. Zooming does not affect the printout, only the preview.

If you want to print with a different **font**, select it from the drop-down list. You can also change the **font size** by selecting one of the common sizes from the drop-down list below the font selection, or by typing in any size between 5 and 99 in that field instead of selecting from the list. If you change the font or the size, EditPad will remember the new font together with the font that you are editing the file in (the font you selected in Options|Font). If you are editing a file in "Arial 10" and print it in "Verdana 12", EditPad will remember that combination. The next time you print a file you are editing in "Arial 10", "Verdana 12" will automatically be selected as the print font. You can change it at any time, of

course. However, if you would use Options|Font to change the editor font to “Courier New 10” before using File|Print, the default printer font would be “Courier New 10” and not “Verdana 12”. If you change the editor font back to “Arial 10”, the default printer font will again be “Verdana 12”.

Press the **Setup button** to access your printer’s setup screen or to select a different printer than the default printer.

“**From page**” and “**to page**” determine the range of pages that will be printed. The value of “from page” must be smaller than or equal to that of “to page”, even if you want to print in reversed order (see below).

Activating “**All pages**” will print all pages in the range specified by “from page” and “to page”, while “**odd pages**” will print only the odd-numbered (1, 3, 5, ...) pages in the range, and “**even pages**” only the even-numbered (2, 4, 6, ...) ones. If the page range is only one page (“from page” equals “to page”) and that page is odd-numbered while “even pages” is activated, or the other way around, nothing will happen when you click the Print button. You will not get an error message.

If you mark “**Reversed**”, the pages will be printed from last to first instead of from first to last. This can be useful if you have an inkjet printer that puts pages in reversed order in the out tray.

Click the “**Headers & Footers**” button to change the headers and footers on the printout.

If you have a monochrome printer or you do not want to waste expensive color ink cartridges, turn off “**syntax coloring**” to print without syntax coloring.

You can change the **line spacing** to any value between 0.8 to 5.0 by typing it into the appropriate field. A few common spacing can be selected from the drop-down list. 1.0 is normal spacing, 1.2 makes the text more readable, 1.5 provides a half line of extra space and 2.0 results in double-spaced text. Line spacing is the amount of space between the top of a line and the top of the next line, expressed in times the total height of the font.

The **margins** can be changed by moving the mouse over one of the dotted blue margin lines. The mouse pointer will change to indicate that you can move the margin, and a hint box will indicate the size of the margin in centimeters and inches. Click on the line, hold the mouse button down and drag the line until it indicates the desired margin position.

Click the “**Print**” button to start printing. Then click the “Close” button or press Escape on the keyboard to close the print preview.

Print Headers

Click the “Headers & Footers” button in the Print Preview window to change the headers and footers to be printed, if any.

The headers and footers are printed inside the margins specified on the print preview. If no headers or footers are specified, their space will be claimed by the text body.

A few special placeholders are available in the headers and footers:

%D Current date, printed using the short date format specified in the regional settings of the Control Panel

%T Current time, printed using the short time format specified in the regional settings of the Control Panel

%FD Date the file was last modified, again printed in short date format.

%FT Time of the day the file was last modified, again printed in short time format.

%FP Full path to the file being printed: folder + filename

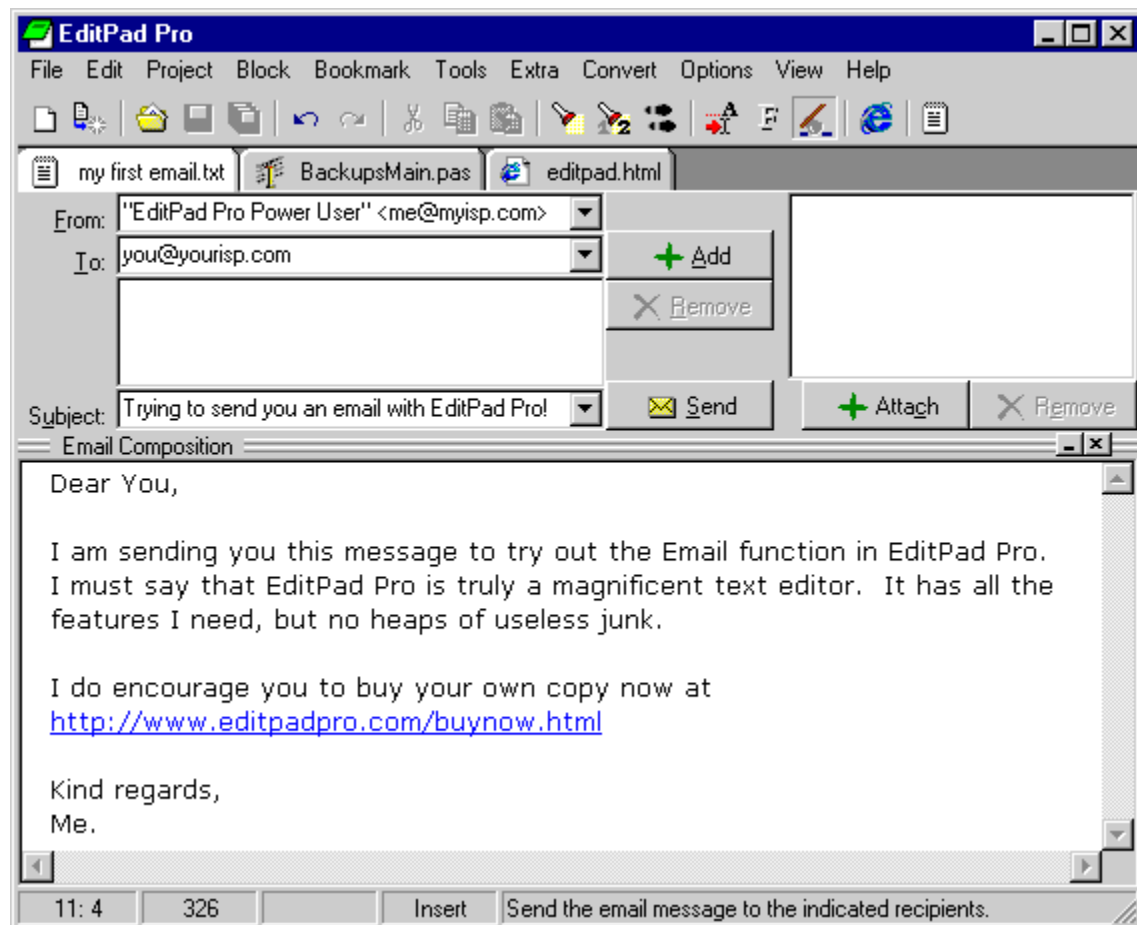
%FN Filename (without foldername) of the file being printed

File|Mail

Select File|Mail from the menu to send the current file to somebody via email.

Before you can use this feature, you need to specify the outgoing mail server that EditPad Pro should use in the Tools Preferences.

The email pane, as you can see in the screen shot below, will be shown.



Type in your own email address in the field labeled **"From"**. If you have previously used the email function, you can quickly select your address from the drop-down list.

Please take care to enter the address in the proper format. First type in your full name (or whatever you want to be called) between double quotes, then type a space followed by your email address between sharp brackets.

Example: "Bill Gates" <bgates@microsoft.com>

Then type in the recipients email address in the **"To"** field. You can either use the same format as for the "from" address, or simply type in the email address by itself.

Example: support@editpadpro.com

Example 2: "EditPad Pro tech support" <support@editpadpro.com>

In the **"Subject"** field, type in the subject of the email message.

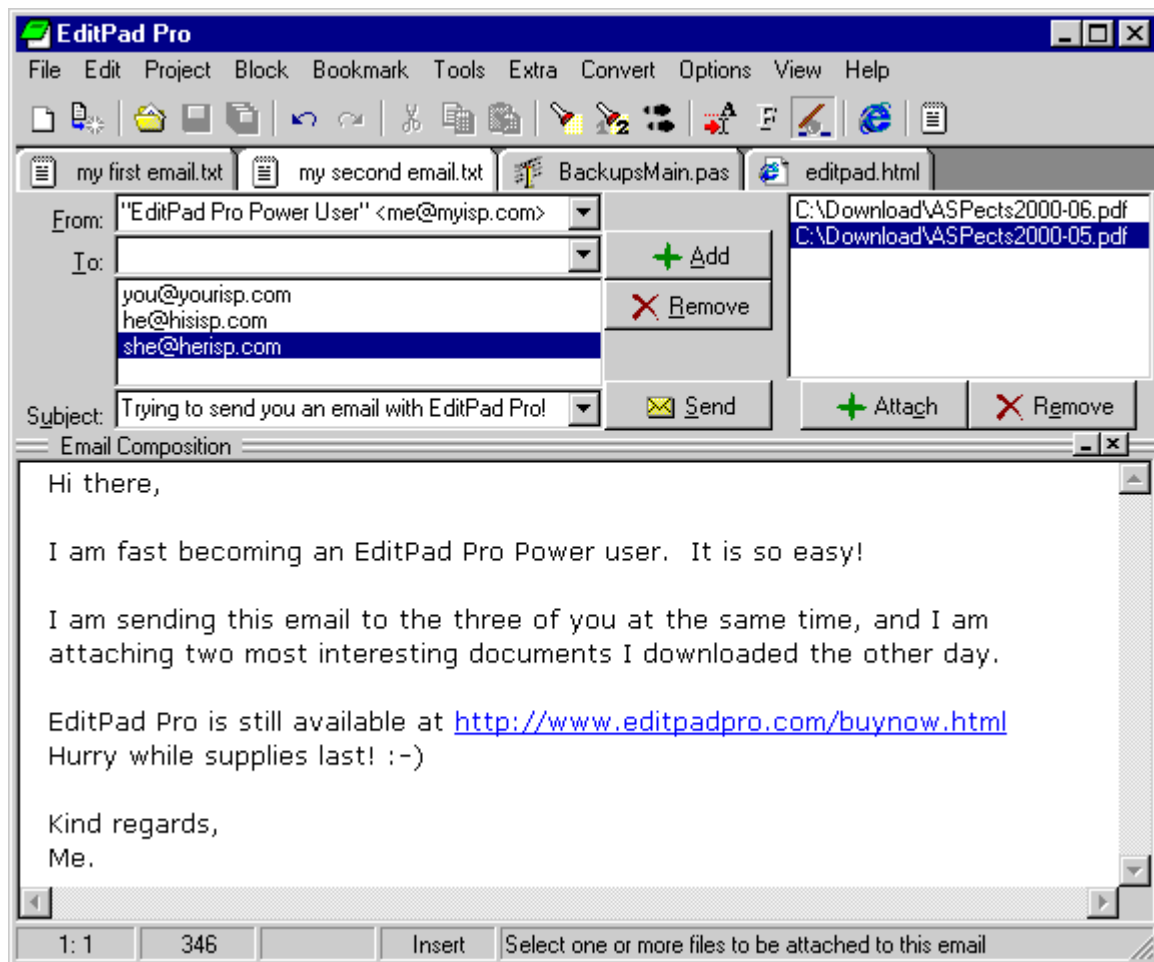
After filling out these three fields, the **"Send"** button will be enabled. Click it to send the email message right away.

File|Mail (advanced)

Select File|Mail from the menu to send the current file to somebody via email.

Before you can use this feature, you need to specify the outgoing mail server that EditPad Pro should use in the Tools Preferences.

Make sure you understand the basic usage of File|Mail first.



If you want to send the same email to more than one address, type in the first address in the "To" field and then click the "Add" button next to the "To" field. Repeat for all additional email addresses.

If you make a mistake, click on the wrong address in the list and then click the "Remove" button.

If you want to add an attachment to the email, click on the "Attach" button. A file open common dialog box will appear. Select the file you want to attach. You can select more than one file, if you want, but remember that most people do not like receiving emails with many or large attachments.

If you make a mistake, click on the mistakenly attached file and then click the "Remove" button.

Click on the "Send" button to send the message.

Sending the attachments may take a while.

File|Reload from disk

Press F4 or select File|Reload from disk from the menu to revert the active file to the status it had when it was last saved.

This can be useful if you are viewing a file that is being written to by another program.

Note that if you switch tabs within EditPad or if you switch to another application and then back to EditPad, it will automatically check if the file on disk has been changed. If so, and you have not modified the file in EditPad, it will be automatically reloaded. If you did modify the file in EditPad, you will be asked if you want to keep the changes made in EditPad or reload from disk.

File|Rename and Move

Use this function to move the active file into a different folder, and/or to change its filename.

This feature is similar to File|Save As, except that the original file will be deleted after the file has been saved under the new name, effectively “moving” the file.

File|Delete

After you confirm the warning message, File|Delete closes the active file and also deletes it from disk.

If Windows keeps a Recycle Bin for the drive the active file is on, it will be moved into the Recycle Bin awaiting its final fate.

File|Close (All)

File|Close closes the active file while File|Close All closes all open files and creates a new one so you can start typing again right away.

If the file has unsaved modifications, depending on the choice you made in Editor Preferences, EditPad will ask if you want to save, automatically save or automatically discard the changes.

To close EditPad itself, click on the X button in the upper right of the Window.

File|Exit

Closes all files and terminates EditPad. This will also remove the EditPad icon that is visible next to the system clock.

If you may be using EditPad again shortly, do not use File|Exit to close it, but click the X button in the upper right corner. This will also close all files and hide EditPad, but EditPad will stay active in the background and continue to show its icon next to the system clock.

Selecting Text

There are several ways to select a piece of text in EditPad.

The most intuitive way is by clicking with the mouse, holding the button down, moving the mouse pointer and releasing the button.

You must start a new selection outside of the current selection, if any, since you will be initialing a drag-and-drop operation otherwise.

If you use the keyboard, hold down the Shift button on the keyboard while moving the text cursor to make the selection.

If you want to make a **rectangular selection** instead of a normal, line-based selection, hold down the Alt button on the keyboard while making the selection as explained above. Note that rectangular selections can only be made when a fixed-width font such as `Courier New` is being used, and word wrap must be off.

To select the entire file at once, use Edit|Select All.

Edit|Undo

Use Edit|Undo to undo the last editing action. Repeat to undo more actions.

You can undo the undo with Edit|Redo if you do so right away. If you take any other action that is remembered by the undo function, the redo list will be cleared. (Unless it is a trivial action such as moving the cursor.)

If you put the undo button on the toolbar, you can undo several actions in one go via the drop-down menu of the undo button on the toolbar. When you select an action, that action and all actions listed above it in the drop-down menu will be undone at once. They will all be added to the redo list too.

The undo feature of EditPad Lite and Pro remembers all changes you made to any file since you opened it in EditPad. Saving a file does not clear the undo list.

Edit|Redo

If you change your mind after picking Edit|Undo, use Edit|Redo to undo it.

The redo function works exactly the same like the undo function, except that making any change to a file in any way except through Edit|Undo or Edit|Redo, will clear the redo list.

In EditPad Lite and Pro, a trivial action such as moving the text cursor or making a selection, will not clear the redo list.

Edit|Cut (Append)

Deletes the current selection and places it on the Windows clipboard. Use Edit|Copy to put the selection on the clipboard without deleting it. The text is always copied in Windows format, regardless of the format the active file has.

If you use Edit|Cut, any data previously held by the clipboard is removed from it. With Edit|Cut Append, the selection is appended to the text already on the clipboard (if it holds text data). Edit|Paste will then paste both the original text plus the selection appended to it.

If you placed the Edit|Cut button on the toolbar, you will see that it has a drop-down menu that initially contains 16 items hexadecimally numbered 0..F. If you select one of these items, the cut function will not use the Windows clipboard but one of the 16 “clipboards” internal to EditPad Pro. These form the EditPad Pro PasteBook. You can paste these using the drop-down menu of the Edit|Paste button on the toolbar.

Edit|Copy (Append)

Places the current selection on the Windows clipboard. The text is always copied in Windows format, regardless of the format the active file has.

If you use Edit|Copy, any data previously held by the clipboard is removed from it. With Edit|Copy Append, the selection is appended to the text already on the clipboard (if it holds text data). Edit|Paste will then paste both the original text plus the selection appended to it.

If you placed the Edit|Copy button on the toolbar, you will see that it has a drop-down menu that initially contains 16 items hexadecimally numbered 0..F. If you select one of these items, the copy function will not use the Windows clipboard but one of the 16 “clipboards” internal to EditPad Pro. These form the EditPad Pro PasteBook. You can paste these using the drop-down menu of the Edit|Paste button on the toolbar.

Edit|Paste

If you select Edit|Paste when the Windows clipboard holds textual data, it will be inserted into the active file at the current position of the text cursor. If the active file is not in Windows format, the text on the clipboard will be automatically converted before it is inserted (the text that stays on the clipboard remains untouched).

If you placed the Edit|Paste button on the toolbar, you will see that it has a drop-down menu that initially contains 16 items hexadecimally numbered 0..F. If you previously used the cut or copy function onto one of these items, you will see the copied text in the drop-down menu. If you select one of these items, the paste function will insert that text instead of the text held by the Windows clipboard.

Edit|Swap with Clipboard

If you select Edit|Swap with Clipboard when the Windows clipboard holds textual data, it will be swapped with the current selection in the active file. That is, the selection is put on the clipboard as happens when you use Edit|Cut and then the text previously held by the clipboard is pasted into the text like Edit|Paste does.

If you placed the Edit|Swap with Clipboard button on the toolbar, you will see that it has a drop-down menu that initially contains 16 items hexadecimally numbered 0..F. If you previously used the cut or copy function onto one of these items, you will see the copied text in the drop-down menu. If you select one of these items, the paste function will insert that text instead of the text held by the Windows clipboard.

Edit|PasteBook

EditPad Pro has 16 internal “clipboards” that form the PasteBook. You can quickly cut, copy and paste using the PasteBook if you add the cut (append), copy (append) and paste buttons to the toolbar. The toolbar buttons have a drop-down menu that provides access to the 16 internal “clipboards” and also shows the beginning of the text held by them.

If you want to modify one of these items, you could, of course, paste it, modify it and then cut into the same item.

However, you can also use Edit|PasteBook to open a separate screen with all the items and edit them there.

First select the item you want to edit from the list on the left and then edit it in the editor at the right.

Edit|Select All

Makes the entire active file the current selection.

Edit|Delete

Deletes the current selection.

Edit|Delete Line

Selects the line the text cursor is currently on, and deletes it.

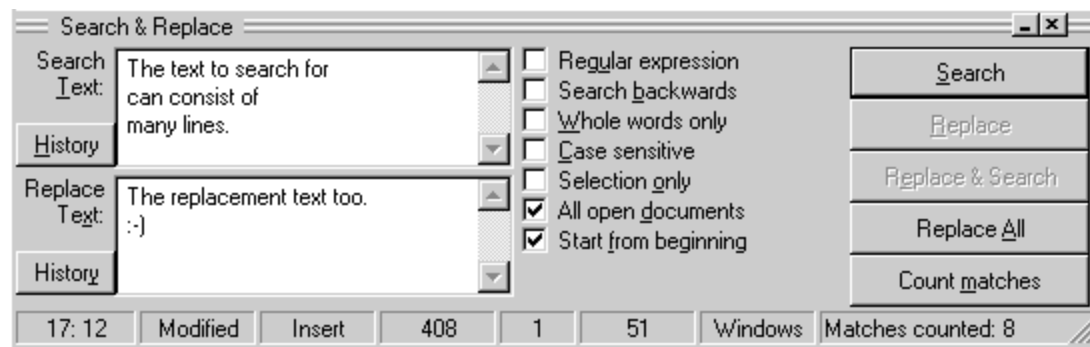
Edit|Search and Replace

EditPad Classic has always been very popular for its powerful search and replace feature. EditPad Pro even improves on it.

When you select Edit|Search and Replace from the menu, the search and replace pane will be displayed at the bottom of the window. You cannot make this pane float, as a floating pane only gets in the way of the things it obscures.

If you want to access the menu with the keyboard while the pane is visible, press Alt, release it, and then press the shortcut character key. If you hold down Alt while pressing the shortcut character, the shortcuts on the search and replace pane take precedence.

You can quickly close the pane by pressing Escape on the keyboard.



Both the search and replace **edit boxes** are full-blown multi-line edit controls, just like EditPad's main editor. Press Shift+Enter on the keyboard to type in and search for or replace with a newline. To type in a tab, press Shift+Tab. To switch from search to replace and back, press Tab. Ctrl+Tab will activate the next file, as usual.

If you right-click on the search or replace edit box, you will see a list of the last 16 search or replace texts you used. Click on one to use it again.

If you want to search using **regular expressions**, first mark the corresponding checkbox. Read the Introduction to Regular Expressions to learn what you can do with regular expressions, or try some of the example regular expressions.

The regex engine used in EditPad Pro is fully compatible with that used by Perl 5, including all the Perl extensions. Backreferences are available in the replacement text through \0, \1, \2, ... \0 is the entire regex match, \1 is the first backreference, and so on.

Note that the regex engine is very strict about newlines, which are in fact a \r\n pair in Windows and not just \n. I strongly advise you to press Ctrl+Enter to type in an actual newline instead of using \r and/or \n. EditPad Pro will automatically replace the actual newline with the appropriate line terminator depending on the current file's type.

Mark **“check backwards”** if you want to search from the end until the beginning. This option is not available in regular expression mode.

Mark **“whole words only”** if only entire words are valid search results. If this option is on, when looking for “cat”, EditPad will not consider the first three letters of “category” a valid search match. Otherwise, it will.

If you turn on **“case sensitive”**, the difference between lowercase and uppercase letters is taken into account and “Dog” will be considered to be different from “dog”. Otherwise, “dog”, “Dog”, “DOG”, and even “doG” are all the same. Turning on this option will greatly improve the speed at which EditPad can do its search, so you may want to do so when searching through a large file.

If you activate “**selection only**”, EditPad will only look inside the current selection instead of the entire file. This does not work with rectangular selections.

You can search through **all open documents** at once too. If the option is on and the search text cannot be found in the current file, EditPad will automatically continue with the next file until all files have been searched. This works with “Replace All” too, making it a powerful function.

If “**start from beginning**” is on, EditPad will start searching from the beginning of the active file instead of the current position of the text cursor. If “all open documents” is on too, “start from beginning” will cause EditPad to activate the first document before it starts searching. “Start from beginning” will be automatically deactivated when you click the “Search” button. If the search cannot find any match, “start from beginning” will be automatically activated.

Click the “**Search**” button to search the next occurrence of the search text. Each time you click “Search” and a match is found, the match counter in the statusbar will increase.

Click the “**Replace**” button to replace the most recently found occurrence of the search text with the replacement text.

Clicking “**Replace and Search**” is the same as first clicking “Replace” and then “Search”.

Click “**Replace All**” to replace all occurrences of the search text with the replacement text. If you do a “replace all” across all open documents and want to undo the action, you have to undo it for each file individually. The undo feature keeps a separate undo list for each file.

Click the “**Count Matches**” button to find all occurrences of the search text and see how many were found. The result will be displayed in the status bar. No modifications are made to any file.

Edit|Search Forward

Pressing F3 or selection Edit|Search Forward is the same as turning off “start from beginning” and “search backwards” and clicking the “Search” button on the search and replace pane.

It will find the first occurrence of the search text after the current position of the text cursor.

Edit|Search Backward

Pressing Shift+F3 or selection Edit|Search Backward is the same as turning off “start from beginning” and turning on “search backwards” and clicking the “Search” button on the search and replace pane.

It will find the first occurrence of the search text before the current position of the text cursor.

Edit|Go to

Asks you for a line number and will then position the text cursor on that line, scrolling if necessary.

Edit|Insert Date & Time

Inserts the current date and time at the position of the text cursor. You can specify which format to use in the Editor Preferences.

About EditPad Pro Projects

An EditPad Pro Project is a series of files that you can open all at once through Project|Open Project.

If you regularly work with a set of related files, it is convenient if you create a project file for them. You could have a project for your web site, for the source code of an application you are working on, the chapters of your next book, etc.

To create a project, open all the files that should be in it, and select Project|Save Project from the menu.

You can then quickly open all those files again later by using Project|Open Project or even Project|Reopen.

Bookmarks are also saved into a project.

Project|Open Project

Opens an EditPad Pro Project saved with Project|Save Project.

When opening a project, all currently open files will be closed first.

Project|Save Project

Saves the list of files currently open and their bookmarks, if any, into an EditPad Pro Project.

Project|Reopen

Shows a list of EditPad Pro Projects recently saved with Project|Save Project, allowing you to reopen them quickly. If you add the Project|Open Project button to the toolbar, the reopen project menu is available as a drop-down menu to the open project button.

When opening a project, all currently open files will be closed first.

At the bottom of the menu, you will see the “Remove Obsolete” and “Remove All” items. The former will remove all files from the reopen menu that no longer exist, the latter will clear the entire reopen menu.

Block|Indent

Indents the current selection by the number of spaces specified in the “block indent” setting in Editor Preferences.

If the “block indent” setting is an integer multiple of the “tab size” setting, and the option “tab inserts spaces” is off, Block|Indent will use tab characters instead of spaces to indent the selection.

If the selection spans more than one line, you can also indent it by pressing Tab on the keyboard. The difference between using Block|Indent and the Tab key is that Block|Indent uses the “block indent” setting while Tab uses the “tab size” setting to determine the amount of indent. If “tab inserts spaces” is off, pressing tab will indent the selection by inserting exactly one tab character at the beginning of each paragraph.

Note that if word wrap is on, only the first line of each paragraph will be indented.

Block|Outdent

Outdents (or unindents) the current selection by the number of spaces specified in the “block indent” setting in Editor Preferences.

If the selection spans more than one line, you can also outdent it by pressing Shift+Tab on the keyboard. The difference between using Block|Outdent and the Tab key is that Block|Outdent uses the “block indent” setting while Tab uses the “tab size” setting to determine the amount.

Block|Comment

Use this feature if you want to comment out a piece of code in a language that does not support multi-line comments.

Block|Comment works with both normal and rectangular selections and will put the comment character(s) in front of each line.

If the current file is being syntax colored by a syntax coloring scheme that contains a character string for single line comments, Block|Comment will automatically comment out that block using those characters. If not, you will be asked to type in the characters you want to put in front of each line in the current selection.

To remove the comment characters, make a rectangular selection that contains them and hit Edit|Delete.

Block|Insert File

Asks you for a file and insert the text it contains at the current position of the text cursor.

Block|Write

Saves the current selection into a file. If the file already exists, EditPad will show a warning and if you confirm, overwrite the file.

If you do not specify an extension for the filename, EditPad will add one depending on the selection you made in the filters (the bottommost drop-down list) in the Block|Write dialog box. If the active filter is “any file” (*.*), no extension will be added. If any other filter is active, the default extension for that file type will be added to the filename. The default extension is the first extension listed in the Extensions setting of the file type’s settings.

Block|Append

Saves the current selection into a file. If the file already exists, EditPad will append the current selection to the end of that file.

Block|Print

Prints the current selection. A preview of the printout will be shown first. It will allow you to make some changes like which font to print with, which pages should be printed. You can also access the printer setup through the preview window.

Bookmark|Go to Bookmark

Positions the text cursor on the line in the active file on which you set the bookmark with the given number through Bookmark|Set Bookmark.

You can quickly jump to any bookmark by pressing Ctrl+1, Ctrl+2, Ctrl+3, etc... on the keyboard.

You can only jump to bookmarks in the active file. If the requesting bookmark does not exist, nothing happens.

Bookmark|Set Bookmark

You can place up to ten bookmarks in every file open in EditPad Pro. You can quickly do so by pressing Shift+Ctrl+1, Shift+Ctrl+2, etc... on the keyboard. When you place a bookmark, you will see a green square with a white number in the left margin, indicating the bookmark's position.

Bookmarks are placed on paragraphs. If a paragraph is broken into several lines by the word wrapping, you will see several bookmark images in the margin, one for each line. You can remove a bookmark by setting the bookmark again on the same paragraph.

Bookmarks are not saved into a file when you use File|Save.

If you want bookmarks to be remembered, use Project|Save Project to save the list of files currently open and their bookmarks. Then use Project|Open Project or Project|Reopen to open the files again, and the bookmarks will be put back at the positions they had when you saved the project.

Extra|Spell Check

Select Extra|Spell Check from the menu to check the spelling of the current file. The spell check function always starts from the beginning of the file.

The pane below appears during the spell check process:



When a word is not found in the dictionary, it will be selected in the editor and reported in the upper left area of the spell check pane. EditPad Pro will start searching through the dictionary for words similar to the misspelled one, and show them in the list. As long as “(searching...)” is visible, EditPad Pro is still looking for more possible, correctly spelled substitutions. You do not have to wait until EditPad Pro is finished. You can correct the error right away.

If the word is really misspelled, you can type in the correctly spelled word in the “**Replace with**” field. You can also click on an item in the list with EditPad Pro’s suggestions and it will automatically be placed in the “Replace with” field. If you click on “**Suggest similar**”, EditPad Pro will start looking for words in the dictionary similar to what you typed into the “Replace with” field.

Then you need to click one of the three replace buttons: “**Replace**” will replace the misspelled word with the replacement just this time. “**Replace All**” will replace the misspelled word with the replacement every time it is found during the current EditPad Pro session. When you pick File|Exit or switch to a different language, the replacement is forgotten. If you click “**Learn Replace**”, EditPad Pro will always automatically replace the misspelled word with the replacement and remember this even if you quit EditPad Pro.

You can also fix the error directly in the editor. After doing so, you will have to click the “**Resume**” button to continue to check the spelling of the rest of the file.

If the word is correctly spelled, you can click “**Ignore**” to accept the word as correct just this time. “**Ignore All**” will make EditPad Pro ignore all occurrences of this word until you switch languages or close EditPad Pro. “**Learn**” will make EditPad Pro add the word to its dictionary and will from then on always be accepted as correct.

If you make a mistake, click on “**Back**” to revert it.

If you switch to the editor to make some changes, you will need to click on “**Resume**” to continue the spell check from where you left off.

Click “**Abort**” to stop and close the pane.

The “**Word List**” button allows you to edit the list of words that you added to the custom dictionary. When you click the button, a new window will appear. The left hand side of the window contains the list of words that you specified as correctly spelt by clicking on the “Learn” button. The right hand side of the window contains pairs of words separated by and equals sign (=). These are the automatic replacements you added by clicking the “Learn Replace” button. Whenever the spell checker finds the word at the left side of the equals sign, it will be replaced with the word at the right of the equals sign.

Extra|Spell Check All

Extra|Spell Check All will activate the first file and start the spell check process. When the first file has been completed, the second file will be activated and the spell checker will be invoked on that file, until all files have been checked or you clicked the Abort button on the spell check page.

See Extra|Spell Check for a description of the spell check pane.

Extra|Sort Alphabetically

If no part of the active file has been selected, Extra|Sort Alphabetically will sort the entire line alphabetically, paragraph by paragraph.

If there is a selection, only the selected paragraphs will be sorted.

A very powerful option is to make a rectangular selection first. The paragraphs that are partially covered by the rectangular selection will be sorted entirely like the selection were a normal one. However, the sort order will not be determined by their entire paragraph text. Instead, the part to the left of the selection rectangle will be temporarily cut off and the paragraphs will be sorted on the remainder of the text.

If you are working with a text files that contains information arranged in columns, padded with spaces, you could make a rectangular selection of the third column and then use Extra|Sort Alphabetically. The data will then be sorted on the third column.

Extra|Next Mark

If you have used Extra|Compare Files, you can use Extra|Next Mark to find the block of differing paragraphs after the text cursor's position.

Extra|Previous Mark

If you have used Extra|Compare Files, you can use Extra|Previous Mark to find the block of differing paragraphs before the text cursor's position.

Extra|Compare Files

If you have two versions of the same text file, you can use Extra|Compare Files to visualize the differences between those files. After that, you can edit the generated difference file to merge both versions into a single, new file.

First you need to open the two files you want to compare. The Extra|Compare Files menu item will be grayed out until you have two or more files open.

To compare the files, first activate the file that contains the original or older version of the document by clicking on its tab. Then select Extra|Compare Files from the menu. In the window that appears, click on the file that contains the newer version of the document, in the list of files that are currently open. The active file is omitted from that list, since you cannot compare a file with itself. The setting for the minimum match size will be explained later. 3 is a good default value.

Click on OK and EditPad Pro will show a progress meter, indicating that it is comparing the files. When it is done, EditPad Pro will have created a new tab that contains the difference output.

The difference is paragraph-based. Paragraphs that are identical in both files are displayed on a normal background. Paragraphs that are present in the old version of the document, are displayed on a red background while those that are only present in the new version, are displayed on a green background. Red is text that has been deleted, green is text that has been added to go from the old version to the new version.

Paragraphs that were not added nor deleted, but have been changed between the two versions are shown like this: the old version of the paragraph is shown in red, followed by the new version in green. If many consecutive paragraphs have been changed, the difference will show a red block with all the old versions of all those paragraphs, followed by a green block with the new versions of all the paragraphs.

The difference output behaves like a regular file in EditPad Pro. You can use any editing command on it and save it for later use.

To merge two versions of the same document, use Extra|Compare files as described above. Then use Edit|Delete Line liberally on the produced output. With Extra|Next Mark and Extra|Previous Mark you can quickly jump from one difference location to the next. Using the keyboard shortcuts Ctrl+Y and F7 will greatly speed up this process.

Minimum number of lines for matching block

When you pick Extra|Compare Files from the menu, EditPad Pro will ask for a file to compare the active one with. Near the bottom of the selection window, you will see the spinner box labeled "minimum number of lines for a matching block".

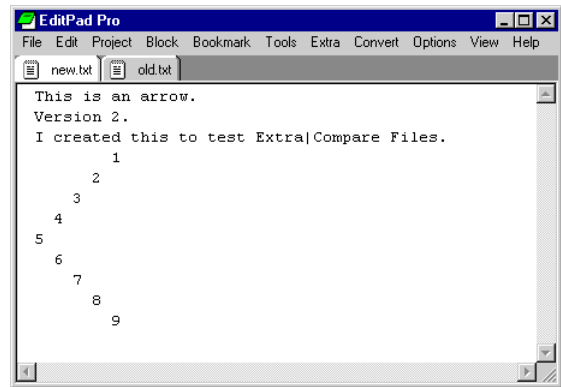
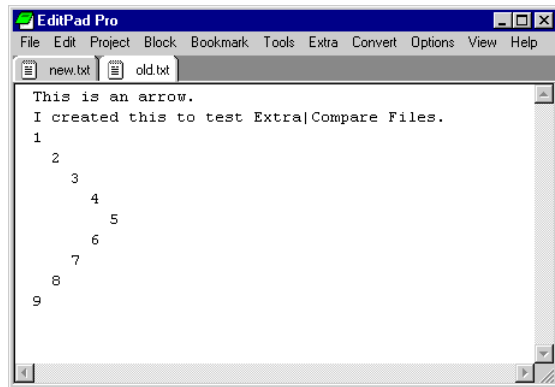
The default setting is 3, but it can be anything between 1 and 9 inclusive.

This value is used by EditPad Pro in the following situation. While comparing the files, it encounters a section of one or more changed paragraphs. In the difference output, these paragraphs are grouped into a single red block showing the original paragraphs, followed by a single green block with the new paragraphs. This is done because all those paragraphs are most likely to be connected somehow: a chapter in a book that was heavily edited, a source code routine that has been modified, etc. Keeping those paragraphs together maintains their logical connection so you can easily see what happened when inspecting the output of Extra|Compare Files.

To even better maintain the logical structure of the document, you can have EditPad Pro also include a few paragraphs in the block that are identical in both versions of the document. This will happen when you set “minimum number of lines for a matching block” to a number greater than one (1).

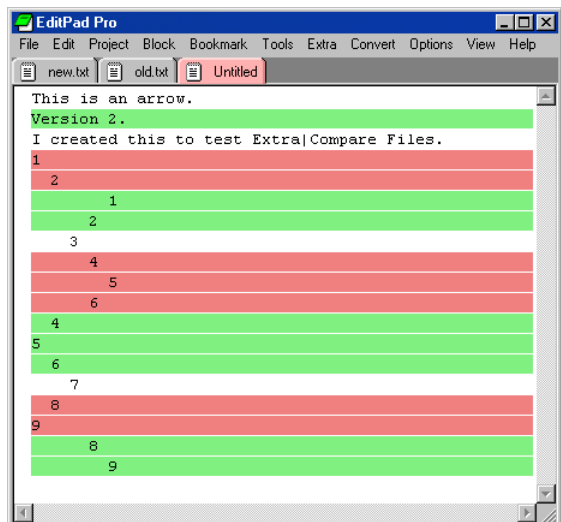
The setting indicates the minimum number of consecutive paragraphs that are identical in both files are required for EditPad Pro to stop grouping the modified paragraphs and add the identical paragraphs as an unchanged block. The graphical illustrations below make this clear.

First you see the old and new version of our test document:



Then we use Options|Compare Files, and set “minimum number of lines for a matching block” to one, effectively turning off the feature. The image at the right shows the result.

You see that EditPad Pro detects the line “Version 2” as being inserted. It also detects that the lines numbered 3 and 7 are present in both files, and that the blocks 1-2, 4-6 and 8-9 have changed. This is technically exact (it is the optimum solution), but logically it makes no sense. The arrow is broken into pieces.



If we try again and set “minimum number of lines for a matching block” to anything greater than 1, we get the result shown by the image below.

Much better! The “I created...” line is still included on its own since it does not follow a block of changed paragraphs, but an inserted one.

However, the lines containing the numbers 3 and 7, even though they are present in both files, are displayed in the middle of the changed block as if they had been changed too. This way, our logical structure, the arrow, is kept together nicely and we can instantly see what happened: the arrow's direction was changed. This was not at all so clear in our first comparison attempt.



Since you will not be comparing arrows but real documents such as source code files, the “minimum number of lines for a matching block” can be configured for the task at hand. Many source code files contain lines with nothing but a curly brace or a

“begin” or “end”. These lines add structure to the source, and not content. When comparing two versions of a piece of source, these lines are likely to be matched all over the place, breaking up the logical structure of the source code in the difference output.

If things don’t look right, experiment with the “minimum number of lines for a matching block” setting.

Extra|Word Count

Select Extra|Word Count from the menu to count the number of words in the current file. Any sequence of the letters a-z and the digits 0-9 is considered a word.

Convert|To Uppercase

Converts the current selection to all uppercase letters.

Convert|To Lowercase

Converts the current selection to all lowercase letters.

Convert|Initial Caps

Turns all the letters in the current selection into lowercase letters, except for the first letter of each word which becomes uppercase.

Convert|Invert Case

Turns all uppercase letters in the selection into lowercase letters, and all lowercase letters into uppercase.

If you have been typing with Caps Lock on, you do not have to start over but let this function come to the rescue.

Convert|To Windows/UNIX/Mac

Convert|To Windows converts the active file to the format used by Windows, which is also used by DOS and Amiga. These operating systems terminate each line with a Carriage Return and Line Feed pair.

Convert|To UNIX converts the active file to the format used by UNIX and derivatives like Linux, BSD and even BeOS. These operating systems terminate each line with a single Line Feed character.

Convert|To Mac converts the active file to the format used by the Apple Macintosh. On the Macintosh, text files use a single Carriage Return character to terminate a line.

Only the line termination characters are converted. Anything else remains untouched.

To EditPad, it does not matter whether a file is in Windows, UNIX or Macintosh format. However, it will most likely matter to any other application that may need to work with the files.

Convert|OEM -> ANSI

Converts the text from the OEM (DOS ASCII) character set to the ANSI character set used by Windows.

If you open a file created with an old DOS program and letters with diacritics and other special characters are wrong, use this function to convert from the OEM character set to ANSI.

Convert|ANSI -> OEM

If the file you are creating needs to be read by an old DOS program, use Convert|ANSI -> OEM before saving it. While this may make the text look bad in EditPad as special characters are converted, the DOS program will know how to deal with it.

Convert|Unicode

Several conversion options are available in the Unicode submenu of the Convert menu.

EditPad cannot edit Unicode files as that would make it incompatible with Windows 95, 98 and ME, which have poor support for Unicode. In Editor Preferences you can specify that whenever you open a Unicode file (UCS-2 or UTF-8), it should automatically be converted into ANSI so you can work on it with EditPad.

If you disabled the automatic conversion option, or if the magic bytes that EditPad looks for at the beginning of the Unicode file to determine the file format were missing, you can use Unicode -> ANSI, Unicode Big Endian -> ANSI and UTF-8 -> ANSI to respectively convert a UCS-2 LE, UCS-2 BE or UTF-8 encoded Unicode file into the ANSI (MBCS) that EditPad can work with.

Note that these Unicode to ANSI conversion functions the current Windows locale. This means that if the Unicode file contains, say, Greek characters, the Greek characters will only be converted properly if the regional settings in the Control Panel are set to Greek. Otherwise, they will be converted into question marks which indicate characters that could not be converted.

If you are creating a file that must be saved in Unicode format, use the ANSI -> Unicode (UCS-2) or ANSI -> UTF-8 conversion commands before saving it. The result will look awful in EditPad, but the file will be saved properly.

Convert|Wrapping -> Line Breaks

When Options|Word Wrap is on, EditPad will break up long lines so they fit within the width of the EditPad window or are no longer than a certain number of characters. This wrapping is dynamic. It is not saved into the file, which will look different if you change the word wrap settings or open it in another application.

If you want to fix a text in its wrapped state, that is, convert all soft line breaks introduced by the wrapping into hard line breaks as if you had pressed Enter at the end of each line, use Convert|Wrapping -> Line Breaks.

This function will also turn off word wrap.

Convert|Tabs -> Spaces

Converts all tab characters in the active file into spaces. This function will only have its desired effect when using a fixed-width font like `Courier New`.

Convert|Spaces -> Tabs

Converts as many spaces into tab characters as possible, without changing the layout of the file. This function will only have its desired effect when using a fixed-width font like `Courier New`.

Convert|ROT-13

Applies ROT-13 character rotation to the current selection, making it a little difficult to read. Applying ROT-13 again restores the original text.

Before the Internet went commercial, people had the good habit to use ROT-13 to conceal possible offensive postings in newsgroups and then add a warning message in plain text. This made sure that nobody would read the text by accident and be offended. If one was offended after applying ROT-13, he or she would only have himself or herself to blame for ignoring the warning.

Convert|Scramble & Descramble

The scramble and descramble option allows you to protect a text with a password. It can be applied to the entire file or the current selection only. If you apply it to the current selection only, make sure that you will be able to identify the selection's boundaries later as you will need to select exactly the same range to descramble the text.

Applying this feature twice with the same password, will restore the original text.

This function is useful for protecting your Christmas shopping list from curious children and similar personal secrets.

Do not use it to protect anything of real value. Your friends and family will not be able to restore the original text without knowing the password, but a government agency or large corporation will surely be able to afford to pay some smart brains to break through the encryption scheme on a rainy afternoon.

EditPad Pro does not contain any strong encryption algorithms since these are considered "munitions" in many countries and thus subjective to strict import and export regulations. EditPad Pro is a text editor, not a weapon.

Options|Auto Indent

Turn on Options|Auto Indent if you want the next line to automatically start at the same column position as the previous line whenever you press Enter on the keyboard while in the editor. EditPad will accomplish this by counting the number of spaces and tabs at the beginning of the previous line and inserting them at the beginning of the new line you created by pressing Enter. This is most useful for writing source code.

Note that “auto indent” is a file type specific setting in EditPad. This means that if you change the setting through the Options menu or the toolbar, it will **not** stick. If you create a new file or open an existing one, the setting made in File Types Preferences for the type of file you are creating or opening, will be used. Also, when you use File|Save As, the setting for the current file will revert to what you specified in File Types Preferences for the file type that corresponds with the file’s new file name.

Options|Number Lines

Turn on Options|Number Lines to have EditPad display a number before each line in the left margin.

In File Types Preferences, you can indicate if you want visible lines or physical lines to be numbered. This setting cannot be changed in the Options menu.

Note that “number lines” is a file type specific setting in EditPad. This means that if you change the setting through the Options menu or the toolbar, it will **not** stick. If you create a new file or open an existing one, the setting made in File Types Preferences for the type of file you are creating or opening, will be used. Also, when you use File|Save As, the setting for the current file will revert to what you specified in File Types Preferences for the file type that corresponds with the file’s new file name.

Options|Paragraph Symbol

Turn on Options|Paragraph Symbol if you always want to see a ¶ character at the end of each paragraph, indicating the invisible line break character(s). Note that if line break characters are selected, the paragraph symbol will always appear to show you that they have indeed been selected.

This menu item will change the “show paragraph symbol at the end of each paragraph” setting in Editor preferences.

Options|Font

If you pick Options|Font from the menu or left-click the font button on the toolbar, a font selection window will appear. Select the font you want to use and click OK. This will change the display font for the active file, but not of all the other files you may have open.

The font button on the toolbar maintains a history of recently used fonts. Click on one to instantly use it.

When you create a new file or open an existing one, the newly opened or created file will use the same font as the active file.

Note that the font setting applies to the entire file, and changes the way it is displayed only. EditPad is a plain text editor and therefore does not support any text formatting.

If you want to edit text files in a language that uses special characters which are not used in Western European languages, you will need to select the correct “script” from the drop-down list near the bottom of the font selection window that appears when you select Options|Font from the menu. If the wrong script is selected, you will see a whole bunch of weird characters instead of readable text. To fix this, simply use Options|Font to select a different script.

Options|Export Preferences

Use Options|Export Preferences to save your preferences into an .ini file. This allows you to transport them to another computer.

Options|Import Preferences

Imports preferences saved with Options|Export Preferences.

View Menu

At the bottom of the View menu you will see a list of files that are currently open. If you click on one of them, that file will become the active one.

If more than 10 files are open, only 10 of them will be listed and the “Show Preceding Tabs” or “Show Following Tabs” menu items will be added. If you click on “Show Preceding Tabs”, the View menu will show the files preceding the ones currently listed. Likewise, “Show Following Tabs” will make the View menu show the next tabs. When you do this, you will need to click on View in the menu bar to open the View menu again.

Preceding files are the files whose tabs are positioned to the left on the tab control, while following tabs are positioned to the right.

View|Next File

Activate the next file.

You can also do this by pressing Ctrl+Tab on the keyboard.

View|Previous File

Activate the previous file.

You can also do this by pressing Shift+Ctrl+Tab on the keyboard.

View|Hexadecimal

Switches between text and hexadecimal editing. Note that the undo and redo lists will be cleared whenever you switch.

View|Character Map

Shows the character map pane.

It allows you to insert any character into the current file by double-clicking on it in the character map.

Click on the X button in the pane’s caption bar to close the character map.

View|Browser

Opens the active file in your computer's default web browser. If the file is modified, it will be saved first.

If this feature does not work, your web browser is not properly configured.

View|New Editor

When you select View|New Editor from the menu, EditPad will split itself in two. You will then have two independent copies of EditPad active. This can be useful if you want to work on two files and want to see both their contents at the same time.

Introduction to Regular Expressions

Regular expressions are used for searching through (usually textual) data. They allow you to search for pieces of text that match a certain form, instead of searching for a piece of text identical to the one you supply. For example, the regular expression `"[0-9]+"` allows you to search through a file for any integer number. If you want, you can read a summary of the history of regular expressions below.

In EditPad Pro, you can use regular expressions to do powerful search and replace operations.

In this introduction, regular expressions will be shown in a `red courier font` and enclosed by double quotes, which are not part of the regular expression you should type in.

The simplest regular expression consists of one literal character. A literal character is a character that has no special meaning, such as: `"a"`. (All special characters are shown below.)

When the regex engine attempts to match `"a"`, it will simply start with the first character, see if it's an `"a"` and if not, continue with the next character until an `"a"` is found. Simple.

The next step is a series of literals, like: `"abc"`. Again, the regex engine will start looking for an `"a"`. When it finds one, it looks for the next token in your regex, which is: `"b"` and attempts to match it with the next character in the text. If it is indeed a `"b"`, it continues again and matches `"c"` to the next character. If again successful, a match has been found and the engine stops.

However, if the character following the `"a"` the engine found is anything but a `"b"`, the engine will "backtrack". That is, the token `"b"` could not be matched, so it continues to look for another match to the previous token, `"a"`. In other words, it starts looking for an `"a"` following the `"a"` it already found. It will continue to do this until either `"abc"` is matched entirely, or there are no more characters in the text left.

It is important that you understand how this backtracking issue works. It is trivial for `"abc"`, but it will not be trivial later on.

Another important concept is that the regex engine is **eager**: it will try to get its work done as quickly as possible and therefore return the first match it finds without looking further if there might not be a "better" (i.e. longer) match later. This means that a regex such as `"Set|SetValue"` applied to the text `"SetValue"` will match `"Set"` and not `"SetValue"`. The regex engine stops as soon as it finds a valid match. To solve this, either rewrite the regex as `"SetValue|Set"` or as `"\b(Set|SetValue)\b"`.

It is very important to understand that the regex engine is **greedy**: it will try to match as many characters as possible. Repeat operators such as `*` and `+` will continue to repeat as long as possible. If the result of this is that the next token cannot be matched, the engine will backtrack and make the `*` or `+` match one character less until either the next token is matched, or the repeater matches nothing in which case the engine will further backtrack. This is why the regex `"\".*)"` will match all the text between the first and the last double quote in the text. The engine will first look for a double quote, then the dot-star will match everything until the end of the text. Since there is no double quote after the text, the regex engine will backtrack and in fact search backwards until it finds a double quote that is not the double quote the first double quote in the regex was matched to. Give this a try in EditPad Pro's search/replace. Then try `"\".*?"` with the question mark turning off the star's greed. Another solution is `"\"^[^"]*"`, which is actually better since it's faster.

Conclusion: the regex engine will always return the leftmost, longest match.

A problem with the backtracking aspect is that if you nest repeat-operators like `*` and `+`, the complexity of the regular expression will quickly reach infinity, most likely crashing the application. Therefore, don't nest repeat operators unless you really know what you are doing (and saved your files first!).

If all of the above made little sense to you, please check out the detailed descriptions of the special characters for some samples. Then read the above text **again** before you start creating your own regular expressions. This will save you a lot of head-scratching.

Characters with special meanings:

.	Matches any character.
[]	Character class.
()	Groups tokens and creates backreferences.
\w, etc	Shorthand character classes.
?	Matches the previous token zero times or one time.
*	Matches the previous token zero or more times.
+	Matches the previous token one or more times.
{ }	Matches the previous token a certain number of times.
^	Matches the concept "beginning of a line" (zero-width)
\$	Matches the concept "end of a line" (zero-width)
	Match the token to the left or the token to the right
\	Escape the next character.
\b	Word boundary.

If you want to use any of the characters above as a literal, you have to escape it with a \

Perl extensions:

?	After a ?, *, + or { }: match as few times (non-greedy) instead of as many times (greedy) as possible.
(?#)	Comment. Anything between the # and) is ignored by the regex engine.
(?:)	Groups tokens without creating a backreference.
(?=)	Zero-width positive look-ahead assertion.
(?!)	Zero-width negative look-ahead assertion.
(?<=)	Zero-width positive look-behind assertion.
(?<!=)	Zero-width negative look-behind assertion.

History of Regular Expressions

In case you wondered where their weird name comes from, regular expressions are named after the mathematical theory behind them. A regular expression engine, a piece of software that tries to match the regex you provide to some text, is an implementation of a series of mathematical formulas. However, as a regex user, you do not need to worry about that.

Regular expressions were first made popular in the 1970's by a UNIX tool called grep. This tool could search through files and folders using a regular expression, and list the results.

The other important milestone in the history of regular expressions is the programming language called Perl, created by Larry Wall. It is available for many platforms, including Microsoft Windows. Perl uses the same regular expression syntax as the old UNIX grep. However, Perl 5 introduced many powerful additions to the regular expression syntax. Regular expressions are an integral part of the Perl language, which has proven to be one of the main reasons why many programmers like to use Perl.

The success of Perl prompted many designers of other programming languages to add support for Perl 5 regular expressions to their language. Popular languages that support Perl 5 regular expressions are Python and PHP. Code libraries for many other languages, like C++ and Delphi, also let developers use regular expressions.

Many applications these days support regular expressions. With PowerGREP, you can search through files and folders on your computer, maintain files with search and replace operations, and collect information from files, all using regular expressions.

Popular text editors, such as EditPad Pro, allow you to use regular expressions in their search and replace features.

If you are a programmer, you can save a lot of programming effort with regular expressions. How often did you write some code to parse a string? With a regular expression, you can do in a couple of lines of code what would otherwise require dozens of lines, and save coding and debugging time. You can easily test the regular expressions in an editor like EditPad Pro or a specialized regular expression tool like PowerGREP, before putting it into your code.

Regular Expressions: . (dot)

The dot matches any character, except for carriage return (`\r`) and newline (`\n`).

With most tools you can mark a checkbox that says “dot matches all” or “dot all” to make the dot match any character, including `\r` and `\n`.

Beware not to use the dot where it is not appropriate. Most people with little experience regarding regular expressions, will try to match a double-quoted string as follows: `"\".*)"` thinking that it will match “anything enclosed by double quotes”.

Actually, this is correct. It will match anything between double quotes, including other double quotes. The `*` is greedy, so `"\".*)"` will match the first double quote, the last double quote and any text in between them, whether that includes other double quotes or not. This is clearly not the way a double-quoted string should be matched.

The solution is either to turn off the star’s greed: `"\".*?)"` or to use a negated character class: `"\"[^\"]*)"`. Both these regexes will properly match a double-quoted string. However, the latter is slightly faster.

If line breaks are not allowed in strings, be sure to turn off “dot all” if using the dot, or to include `\r` and `\n` in the negated character class: `"\"[^\r\n]"`.

If double quotes are allowed in strings if they are escaped with a backslash, things get more complicated. The best solution is: `"\"[^\\"\\\\]*(\\\\.|[^\\"\\\\])*"`. If you want to know why this is the best solution, read Jeffrey Friedl’s excellent book [“Mastering Regular Expressions”](#) (a.k.a. the Hip Owls book, ISBN 1565922573). Note that this regex will allow the quoted string to span multiple lines. If you don’t want this, use `"\"[^\\"\\\\\r\n]*(\\\\.|[^\\"\\\\\r\n])*"` instead.

Regular Expressions: []

A character class matches a single character that is present in the character class.

`"b[aei]d"` will match "bad", "bed" or "bid", but not "bod", "bzd" or "b@d".

The character class is a single token, so you can easily repeat it like in `"[bo]+"` which will match "b", "o", "bo", "ob", "bob", "bboobboboo" or any combination of b and o characters.

You can match a range of characters using the hyphen: `"A[3-6]"` will match A3, A4, A5 and A6 but not A2 or A7. A character class can contain many ranges: `"[A-Za-z]+"` matches any series of alphabetic characters, uppercase or lowercase. You can also put in single characters together with the ranges: `"[$_A-Za-z0-9]"` matches any letter, number, dollar sign or underscore.

You can negate a character class by typing a caret right after the opening bracket: `"a[^a-z]"` will match a\$, aA, a0, a@ but not aa, ab or az. Important to note is that it will also not match an a on its own. A negated character class matches a character that is not present in the character class. The above regex matches "an a followed by a character that is not a lowercase letter". If you want to match "an a not followed by a lowercase letter", use a zero-width assertion: `"a(?:[a-z])"`. This regex will match an "a" (and only the a, not including any following characters) that is not followed by another lowercase letter.

Remember that the following characters have a special meaning in a character class: the backslash (\), closing bracket (]), caret (^) and the hyphen (-). If you want to use them as characters in a character class, you have to take extra care. If you want to include a backslash, precede it with another backslash: `"[\\a-z]"`. To include the closing bracket, precede it with a backslash: `"[\\]a-z]"`. To include the hyphen, place the hyphen right after the opening bracket or the negation caret: `"[-a-z]"`. To include a caret, put it anywhere except right after the opening bracket: `"[a-z^]"`.

Note that all other characters are ordinary characters inside a character class. Characters like the dot (.) or the star (*) that have a special meaning outside character classes, do not have that special meaning inside a character class. So there is no need to escape them with a backslash. To match a plus sign or a dot, simply use: `"[+.]"`.

Note that `"[a]"`, while perfectly valid, makes no sense. Just use `"a"`.

Regular Expressions: ()

To group a series of tokens, simply put them inside parenthesis: `"(abc)+"` will match abc, abcabc, abcabcabc, and so on.

Parenthesis also create a backreference. To use the first backreference, type `\1`, for the second, use `\2`, third is `\3`, and so on.

Example: `"([a-z])a\1"` matches "dad", "zaz" and even "aaa"; that is: any sequence of three lowercase characters of which the middle one is an a.

How this works? The first token the regex engine encounters is `[a-z]` which matches any lowercase character. Since this token is inside parenthesis, it is put in backreference 1. The second token is `a`, a literal, and the third token is `\1` which contains the character that was matched by the `[a-z]`. The `\1` is always treated as a literal.

To figure out the number of a backreference, simply count the opening parenthesis in your regular expression. The first opening parenthesis starts the first backreference, the second (is \2, and so on. Note that it doesn't matter if a part of the regex has actually been matched. Those backreferences will simply remain empty.

Also, be careful when using backreferences together with repeat operators. If the regex "`([abc])+`" matches "cbaabb", \1 will contain "b" which is the last character matched by the character class it contains. However, if the regex "`([abc]+)`" matches "cbaabb", \1 will contain "cbaabb" as well.

If you want to use parenthesis for grouping, but do not want to start a new backreference, use "`(?:abc)`" instead.

Regular Expressions: ? (Zero or One)

If a token is followed by a question mark, that token will be matched either once or not at all.

For example: "`ab?c`" will match "abc" and "ac". Note that the question mark is greedy, so abc will be attempted first, and then ac. This doesn't matter in this sample, but it could matter in some cases. Put another question mark behind the question mark to turn off the greed.

To apply the question mark to a series of tokens, put those inside parenthesis: "`ab(cd)?ef`" will match "abcdef" and "abef".

Regular Expressions: *

If a token is followed by an asterisk, it will be repeated zero or more times.

Example: "`ab*c`" matches "ac", "abc", "abbc", "abbbc", and so on.

Note that the * is greedy. This is important if the token being repeated also matches what the token following the star would match. Put a question mark after the star to turn off its greed.

Regular Expressions: +

If a token is followed by an plus sign, it will be repeated one or more times.

Example: "`ab+c`" matches "abc", "abbc", "abbbc", and so on.

Note that the + is greedy. This is important if the token being repeated also matches what the token following the plus would match. Put a question mark after the plus to turn off its greed.

Regular Expressions: { }

Use curly braces to indicate that a token should be repeated a certain number of times.

`"ab{5}c"` will match `"abbbbbc"` and nothing else.

`"ab{2,5}c"` will match `"abbc"`, `"abbbc"`, `"abbbbc"` and `"abbbbbc"`.

`"ab{1,}c"` is the same as `"ab+c"`.

`"ab{3,}c"` matches `"abbbc"`, `"abbbbc"`, and so on with no upper limit to the number of bs, but with a minimum of 3.

`"ab{0,3}c"` matches `"ac"`, `"abc"`, `"abbc"` and `"abbbc"`.

Again, the braces are greedy and will try to repeat as often as they can, and then backtrack. Put a question mark after the closing brace to turn off the greed.

Regular Expressions: ^

The caret matches the concept "beginning of a line". It does not match any actual characters.

Example: `"^ *[0-9]+"` matches any number at the beginning of a line, possibly preceded by some spaces.

Regular Expressions: \$

The dollar sign matches the concept "end of a line". It does not match any actual characters.

The regex `". $"` matches the last character on a line, whatever that character is.

The regex `"#.*$"` matches a hash sign and any characters that follow it until the end of the line, a single-line comment in many languages.

Regular Expressions: |

Matches either the token to the left or the token to the right of the pipe.

This token has the lowest precedence of all regex tokens, so you may need to use paranthesis.

`"ab|cd"` will match `"ab"` and `"cd"`.

`"a(bb|cc)d"` will match "abbd" and "accd".

If you have many options to choose from, just use many pipes:

`"a(bc|cd|qq)d"` will match "abcd", "acdd", and "aqqd".

Remember that the regex engine is eager. Tokens separated by pipes are processed from the left to the right. As soon as a match is found, the entire pipe group is considered to have matched and the following tokens will no longer be evaluated. This means that a regex such as `"Set|SetValue"` applied to the text "SetValue" will match "Set" and not "SetValue". The regex engine stops as soon as it finds a valid match. To solve this, either rewrite the regex as `"SetValue|Set"` or as `"\b(Set|SetValue)\b"`.

Note that `"a|b|c"` (single literal characters) makes no sense. Use the character class `"[abc]"` instead.

Regular Expressions: \

If you want to use any of the special characters `.` `[]` `()` `?` `*` `+` `{ }` `^` `$` `|` `\` as literals in your regular expression, you need to precede them with a backslash (`\`). To match a word enclosed by braces, use `"\{\w+\}"`.

Special literals:

`\n` Line Feed (LF)

`\r` Carriage Return (CR)

`\t` Tab

`\xxx` Character, XX is a hexadecimal value between 00 and FF.

The backslash is also used for shorthand character classes, word boundaries, and backreferences. These are discussed elsewhere in this introduction.

Regular Expressions: \b

`\b` matches the non-character between a word character, and a non-word character. A word character is a character that can be used to form words. This not only includes the letters a-z, but also letters with diacritics such as é. A non-word character is any character that is not a word character.

`\b` can be used to perform a "whole words only" search. `"\bcat\b"` will match the word "cat", but not the first three letters in the word "category". The simple regex `"cat"` would match the first three letters in "category".

To match a list of reserved words, you could use:

`"\b(and|else|if|not|or|then|xor)\b"`

The effect of putting the entire regex between `\b`'s is like turning on "whole words only" in a normal (non-regex) search and replace.

Regular Expressions: \w, \s, \d, etc...

Shorthand notations are available for certain character classes. You can use them on their own, or as part of a character class.

`"\d"` and `"[\d]"` are equivalent to `"[0-9]"`

`"[\da-fA-F]"` is equivalent to `"[0-9a-fA-F]"`. Both match a hexadecimal character.

Note that any of these shorthand character classes will only match a single character, just like regular character classes using the square bracket notation. To match an integer, use `"\d+"`. To match a hexadecimal number, use `"[\da-fA-F]+"`.

`\d` matches the digits 0-9

`\D` matches any character that `\d` does not match (i.e. a non-digit)

`\w` matches a word character. A word character is a character that can be used to form words. This not only includes the letters a-z, but also letters with diacritics such as é. If your computer supports non-Latin scripts, `\w` will also match word characters in those scripts.

`\W` matches any character that `\w` does not match (i.e. a non-word character).

`\s` matches the whitespace characters space and tab. It is equivalent to `"[\t]"` (notice the space).

`\S` matches any character that `\s` does not match.

Make sure not to confuse the uppercase shorthands with negative zero-width assertions. The uppercase shorthands are negated character classes. `\D` is short for `"[^d]"` which is short for `"[^0-9]"`.

Regular Expressions: Zero-Width Assertions

Suppose you want to do the following: "Find me a single character a. The a must be followed by any lowercase letter that is not an a".

This looks easy: `"a[b-z]"` will do the trick.

Wrong! This regex will match the a and the character that follows it, like "ab", "ad" or "az", while we only want the a.

Solution: use zero-width positive look-ahead assertion: `"a(?!a)"`. This will match an "a", and only the "a", that is followed by any lowercase letter but another a. The `(?!a)` part matches the regex it contains, but then discards its results, only returning "OK" or "not OK".

There are four kinds of zero-width assertions:

`(?=)` Zero-width positive look-ahead assertion.

`(?!)` Zero-width negative look-ahead assertion.

`(?<=)` Zero-width positive look-behind assertion.

`(?<!=)` Zero-width negative look-behind assertion.

Example: you want to match the 7th non-whitespace character on any line: `"(?<=^\S){7}\S"`.

How this works: the first token is a positive look-behind assertion. This will be initially skipped over by the regex engine. The next token is `\S`. The regex engine will look for the first non-whitespace character and then check if the look-behind assertion is OK. It will be OK if, before the position where `\S` matched, it can find the beginning of a line and then exactly 7 characters.

Another example: we want to match a single-line PHP comment. In PHP, a language for creating dynamic web pages, a single-line comment is started with `//` or `#` and ends at the end of the line. However, it can also be terminated by the `?>` tag which terminates the PHP script altogether (and continues with HTML). The `?>` tag is definitely not part of the comment, so we cannot just put in: `"(//|#).*?(\\?>|$)"` as the `?>` will then be part of the regex match. Instead, we need to use zero-width positive look-ahead: `"(//|#).*?(?=\\?>)|$)"`. (`?` is a special character, so we escape it with `\\`)

The difference is very important. Suppose you want to do a search/replace to delete all comments from your PHP script. With the former regex, you may be removing some `?>` tags also, which is clearly undesired as it will break your script. The zero-width assertion solves this nicely. It still checks for the presence of the closing tag, but does not make it part of the regex match.

Example Regular Expressions

Since the Introduction to Regular Expressions may be a little dry, you may want to try these examples that show the power of EditPad Pro's search and replace feature when using regular expressions.

- Building a Regex to Match a Floating-Point Number
- Remove Trailing Spaces
- Unwrapping Text
- Removing Duplicate Lines
- Keeping only Duplicate Lines
- Removing Comments

Building a Regex to Match a Floating-Point Number

In this example, I will show you how you can avoid a common mistake often made by people inexperienced with regular expressions.

As an example, we will try to build a regular expression that can match any floating-point number. Our regex should also match integers, and floating point numbers where the integer part is not given (i.e. zero). We will not try to match numbers with an exponent, such as 1.5e8 (150 million in scientific notation).

At first thought, the following regex seems to do the trick: `"[-+]?[0-9]*\.[0-9]*"`

This defines a floating point number as an optional sign, followed by an optional series of digits (integer part), followed by an optional dot, followed by an optional series of digits (fraction part).

Spelling out the regex in words makes it obvious: everything in this regular expression is optional. This regular expression will consider a sign by itself or a dot by itself as a valid floating-point number. In fact, it will even consider an empty string as a valid floating-point number. While this doesn't matter in EditPad Pro (it ignores zero-length matches), this regular expression can cause serious trouble if it is used in a scripting language like Perl or PHP to verify user input.

Not escaping the dot is also a common mistake. A dot that isn't escaped will match any character, including a dot. If we hadn't escaped the dot, 4.4 would be considered a floating point number, and 4X4 too!

When creating a regular expression, it is more important to consider what it should NOT match, than what it should. The above regex will indeed match a proper floating-point number, because the regex engine is greedy. But it will also match many things we don't want, which we have to exclude.

Here is a better attempt:

`"[-+]?([0-9]*\.[0-9]+|[0-9]+)"`

This regular expression will match an optional sign, that is either followed by zero or more digits followed by a dot and one or more digits (a floating point number with optional integer part), or followed by one or more digits (an integer).

This is a far better definition. Any match will include at least one digit, because there is no way around the `"[0-9]+"` part. We have successfully excluded the matches we don't want: those without digits.

We can optimize this regular expression as:

`"[-+]?([0-9]*\.)?[0-9]+"`

If you also want to match numbers with exponents, you can use:

`"[-+]?([0-9]*\.)?[0-9]+([eE][-+]?[0-9]+)?"`

Notice how I made the entire exponent part optional, rather than making each element in the exponent optional.

Remove Trailing Spaces

To remove the trailing spaces from every line in the current file, type in the following regular expression in the search box:

“ `+$` ”

That is a space (), a plus (+) and a dollar (\$).

Leave the replace text empty, make sure the “regular expressions” checked and hit replace all.

The space is a literal in regex syntax, the plus means “the previous character one or more times” and the dollar means “end of line”. Note that the dollar does not match actual characters, only the concept “end of line”. So our complete regex will match a series of one or more spaces before the end of a line. The + is greedy, which means it will try to gather as many spaces as possible, so all trailing spaces will be removed.

Unwrapping Text

This regular expression does the opposite of the Convert|Wrapping->Line Breaks function. It removes all line breaks, except those that insert an empty line in the text. This can be useful if you want to create a nice printout of an email message. Emails are typically wrapped at 72 characters or so. If you print them with a proportional font like Arial, there will be a lot of white space on the right hand of the printed page. Removing the line breaks first gets rid of this white space. Try it yourself.

First clear the search box and then press Shift+Enter to put a line break into it. On the second line in the search box, type: `([^\r\n=-])`

The round brackets create a backreference that we’ll use later. The square brackets indicate a character set, which is negated by the caret (^). \r and \n are the codes for carriage return and line feed.

This regex will effectively match a line break followed by any character that is not a line break character, a = or a -. This character is stored into the first backreference.

We don’t want the = or - since lines beginning with those characters start underlining in our sample text (this newsletter), and we do not want to unwrap those. You may want to specify other characters tailored to the text you are unwrapping.

As the replace text, type in: `\1`

Because the regex matches the line break plus an extra character, and we only want to delete the line break, the replacement text must put the extra character back in.

Make sure “regular expressions” and “start from beginning” are on, and hit replace all.

Removing Duplicate Lines

With EditPad Pro it is easy to sort a text file alphabetically. But having done that, suppose that you then want to remove the extra copies of any repeated lines. That can be done with regular expressions, as follows:

1. In the Edit menu, click on "Search and Replace".
2. Tick the "Regular expression" and "Start from beginning" boxes. Make sure to leave the other boxes clear.
3. On the first line of the "Search Text" box key in the following line: `^(.*) (`
4. Key in Shift+Enter, which will move the cursor to the second line.
5. Key in the following line: `\1)+$`
6. The "Search Text" box should now look like this:
`^(.*) (`
`\1)+$`
7. On the first line of the "Replace Text" box key in `\1`
8. Click on "Replace All"

Keeping Only Duplicate Lines

This regular expression example does the opposite of the Removing Duplicate Lines example.

Instead of deleting each duplicated line in a sorted file (keeping one copy of duplicated lines), this example will delete any line that does not appear at least twice. If a line appears n times, it will appear n-1 times after using the sample below. So if you use this regular expression twice, you will keep only those lines that appear three times.

Here is how to keep only duplicated lines:

1. In the Edit menu, click on "Search and Replace".
2. Tick the "Regular expression" and "Start from beginning" boxes. Make sure to leave the other boxes clear.
3. On the first line of the "Search Text" box key in the following line: `^(.+) (`
4. Key in Shift+Enter, which will move the cursor to the second line.
5. Key in the following line: `(?!\\1$)`
6. The "Search Text" box should now look like this:

`^(.+) (`
`(?!\\1$)`

7. Make sure the “Replace Text” box is completely empty.

8. Click the “Replace All” button.

Only lines that appeared twice or more times will remain.

This can be very useful if you want to see which lines appear in two files:

First, open both files, sort them alphabetically and remove any duplicate lines from both files.

Then use copy/paste to copy the second file into the first one, and use Extra|Sort Alphabetically to sort the combination.

Now apply the search and replace operation described above to the combination file. Only lines that appeared twice in the combination, and thus in both original files, will remain.

Removing Comments

In this example, I will show you how to remove all comments from a C++ or Java source code file. You can easily modify the example regular expressions to make them work on the comment syntax used by your favorite programming language.

First we will remove all single-line comments. To do this, we use a regular expression that matches a pair of forward slashes and anything up to the end of the line. Type this in the search box: `//.*$`

Note that this regular expression is not particularly intelligent. If you would happen to have a pair of forward slashes in a character string, they will be removed just the same. So be careful.

Leave the replace box empty, make sure “regular expressions” and “start from beginning” are on, everything else is off and hit Replace All.

Now the multi-line comments, delimited by `/*` and `*/`. Since the star is a special character, we escape it with a backslash. Type this in the search box: `/\.*.**/`

This looks complicated, but it is not. It’s just `.*` between the comment delimiters. Again, if your source code would happen to contain `/*` or `*/` inside a character string, the regular expression will match it as a comment just the same and the replace all will remove it.

Leave the replace box empty, make sure “regular expressions” and “start from beginning” are on, everything else is off and hit Replace All.

EditPad Pro License Agreement

EditPad Pro is copyright © 1996–2002 Jan Goyvaerts. All rights reserved.
“EditPad” and “JGsoft - Just Great Software” are trademarks of Jan Goyvaerts.

IMPORTANT - READ CAREFULLY

This license statement and limited warranty constitutes a legal agreement (“License Agreement”) between you (“Licensee”, either as an individual or a single entity) and Jan Goyvaerts (“Vendor”), owner of the brand “JGsoft - Just Great Software”, for the software product EditPad Pro (“Software”) of which Jan Goyvaerts is the copyright holder.

The Web Site referred to in this agreement is <http://www.editpadpro.com/>

BY INSTALLING, COPYING, OR OTHERWISE USING THE SOFTWARE, YOU AGREE TO BE BOUND BY ALL OF THE TERMS AND CONDITIONS OF THE LICENSE AGREEMENT.

Upon your acceptance of the terms and conditions of the License Agreement, Jan Goyvaerts grants you the right to use the Software in the manner provided below.

If you do not accept the terms and conditions of the License Agreement, you are to promptly delete each and any copy of the Software from your computer(s).

This license agreement only applies to the software product “EditPad Pro” and not to any other product even if that product is similar to EditPad Pro and has a similar name.

The Vendor reserves the right to license the same Software to other individuals or entities under a different license agreement.

After accepting this license agreement, the Licensee is permitted to use the Software under the terms of this agreement for no more than thirty (30) days, and for evaluation purposes only, without payment to the Vendor. For this purpose, the Vendor provides a special free trial version of the Software that can be freely downloaded from the Vendor's web site.

If the Licensee wishes to use the software for more than thirty (30) days and/or for purposes other than evaluating the Software, the Licensee must purchase a single user license or a multi-user license from the Vendor. This license bears the name of the licensed person or entity and is not transferable to any other party. Pricing and availability is subject to change without prior notice. The Licensee can consult the most recent pricing information at <http://www.jgsoft.com/pricing.html>

If a single user license was bought, the Licensee has the option of installing the Software on a single machine possibly used by one or more persons, or installing the Software on several machines used exclusively by a single person. Any combination of these options, or installing the Software on a network server, is not permitted.

If a multi-user license was bought, the Licensee may choose to install the Software on as many machines as licenses were bought, but no more, or the Licensee may choose to install the Software on one or more network servers on the condition that the number of client machines that have access to that server or those servers, is equal to or less than the number of licenses bought. When the number of client machines grows, the Licensee must buy additional licenses.

Regardless of the number of licenses bought, the Licensee has right to place an additional copy of the Software and the license key on a removable medium for backup purposes to protect the investment made.

Technical support is available directly from the Vendor at no additional charge. When encountering problems, the Licensee must first visit the Web Site and read the information presented there to make

sure the question or problem is not already answered there. If not, the licensee may write to technical support email address with his question in either the English or Dutch language.

The Software is provided "as is". In no event shall the Vendor or any of his affiliates be liable for any consequential, special, incidental or indirect damages of any kind arising out of the delivery, performance or use of this Software, to the maximum extent permitted by applicable law. While the Software has been developed with great care, it is not possible to warrant that the Software is error free. The Software is not designed or intended to be used in any activity that may cause personal injury, death or any other severe damage or loss.

When errors are found in the Software, the Vendor will release a new version of the Software that no longer contains those errors a reasonable amount of time after the Vendor is given an accurate description of those errors. Which amount of time is reasonable will depend on the complexity and severity of the errors. The Vendor will mention the release on the Web Site and, at the Vendor's option, directly contact the Licensee to announce the new release. The Licensee can then, at their option, upgrade to the latest version or to continue to use the older version the Licensee already has. In either case, no payment to the Vendor is required. In the latter case, the Licensee will no longer be entitled to technical support until the Licensee has upgraded to the latest version.

The Vendor reserves the right to charge an upgrade fee in the case of major new enhancements or additions to the Software. This major new version will then start a new version line that will use version numbers clearly distinguishable from the old version line. The Licensee has no obligation to upgrade to the new version line and the Vendor will continue to make available the latest version of the previous version line and release new versions in the old version line in the case errors are still found, and provide technical support for it.

You must not attempt to reverse compile, modify, translate or disassemble the Software in whole or in part. You must not run the Software under a debugger or similar tool allowing you to inspect the inner workings of the Software.

The Software remains the exclusive property of the Vendor. Any Licensee who fully complies with the terms in this license agreement may use it according to the terms of this license agreement. You must not give copies of the Software or your license key to other persons or entities. You must not transfer the Software or your license key to another person or entity. You must also take reasonable steps to prevent any third party from copying the software from one of your machines without your permission.

You may distribute the evaluation version of the Software that is available for public download on the Web Site at the moment that you do distribute it, on the condition that you do this by making identical copies of the downloaded file(s). Public download means any file that can be downloaded by browsing to the Web Site and navigating through the links visible on the page, without the use of any password or identification that you may type in or that may be automatically supplied by your browser if you have typed it in before.

You must not ask payment for the act of distributing the evaluation version of the Software or for the evaluation version itself. You may ask a reasonable contribution to cover your expenses in material, shipping and communication costs. You must make it clear to the recipient that you are sending an evaluation version and that the recipient will have to accept a license agreement in order to evaluate it, and make payment in order to fully use the Software. You must not distribute the evaluation version by making it part of a larger package, unless that package is a collection of evaluation software and other software that does not require payment.

The Vendor reserves the right to revoke your license if you violate any or all of the terms of this license agreement, without prior notice.