

## Introduction



v3.5

Copyright (c) 1999-2002 Xarka Software

---

OptiPerl will help you create CGI scripts in Perl, offline in Windows. It is a fully integrated developing environment for creating, testing, debugging and running perl scripts, directly or through associated html documents.

Read [Setting Up](#) to setup Windows and OptiPerl.

*Features include:*

- Offline editing of CGI Perl Scripts.
- Complete emulation of a real server - scripts can be run indirectly from html documents.
- Live preview of the scripts in the internal web browser.
- Feature packed editor with syntax highlighting.
- Completely integrated debugging with live evaluation of expressions, watches, breakpoints, flow control.
- Code librarian and code templates.
- Context sensitive help on core perl and module documentation.
- Query editor to easily create Queries to be tested with a GET and POST method.
- Many tools like URL Encoder, Perl Printer, Pod Viewer and others.
- Projects to organize and publish a set of scripts.
- Version converter to handle non supported perl functions in windows
- Script uploader to FTP servers.
- Printing script and exporting as HTML with syntax highlighting.
- File explorer.

OptiPerl is not Free! Some features have been disabled in this demo version. However for a small amount of money payable on-line with your credit card, you may download the full featured version.

*Why should you buy this program? Well, consider the following:*

- OptiPerl is a great learning tool. You can use it to learn Perl and CGI programming in the comfort of a visual editor, without loss of time and money because of loading from dos perl or being for hours on the internet.

*So if you program for hobby or want to introduce yourself in a pleasant way to CGI programming, OptiPerl is for you.*

- Professionals designing perl programs and cgi scripts use it. You can save much time by quickly creating, debugging and testing all the ideas you have increasing productivity.

*For novice or advanced users, for hobby or professional use, OptiPerl is bundled with all the tools you need to make CGI in Perl Programming easy!*

---



**OptiPerl is Copyright (C) 1999-2002 by Xarka Software**

Portions Copyright (C) 2000 RITLABS S.R.L.

Perltidy is Copyright (C) 2000, 2001 by Steven L. Hancock

Package YAPE::Regex::Explain is work of Jeff "japhy" Pinyan, CPAN ID: PINYAN

Package Text::Balanced is work of Damian Conway, CPAN ID: DCONWAY

Examples are from the book "Official Guide to Programming with CGI.pm" by Lincoln D. Stein - John Wiley & Sons, Inc. - ISBN 0-471-24744-8 - <http://www.wiley.com/compbooks/stein/index.html>

*Many thanks to **David J. Marcus** and **J. Walker** for a very detailed discussion on improving OptiPerl.*

## **Registering**

Registering will enable you to:

- Receive the full version of OptiPerl.
- Enable editing, debugging and running of unlimited sized scripts.
- Get all future upgrades. You will receive a password (that can be retrieved by e-mail if ever lost) to login to your personal update page which will always have the latest version, your registration info and any related files.
- Receive with priority technical support on OptiPerl.
- Get the full documentation of perl and apache, indexed and searchable.

*Not convinced that you need it?*

Please read the [introduction](#) to see how you can use it.

**Or view OptiPerl's features [Optically](#).**

We are introducing price differentiation so you can enjoy this program at a price that fits your budget:

- OptiPerl Standard
- OptiPerl Professional
- OptiPerl Commercial

Read more in OptiPerl's homepage about the differences between the versions.

All registered customers will receive free of charge future updates.

You can register with a credit-card worldwide. Accepted currencies are US Dollars and Euros. You can also choose from other means of payment like a bank transfer or a check. Follow the direction on the order page for more information.

OptiPerl Homepage:

**<http://www.xarka.com/optiperl/>**

E-Mail

[optiperl@xarka.com](mailto:optiperl@xarka.com)

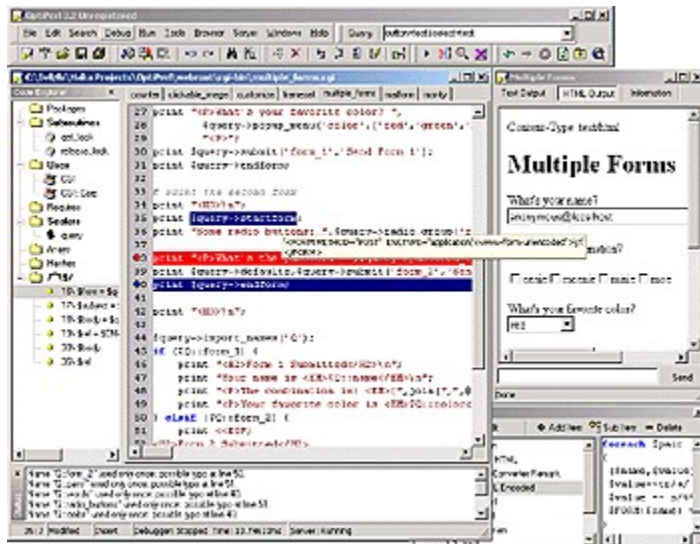
## Features Optically



### Features

---

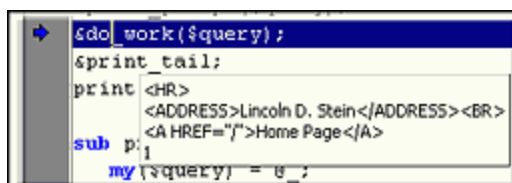
## Integrated Development



OptiPerl is the complete developing solution for Perl. Scripts can be created, edited, debugged and run all offline in a Visual Environment. Internal Web Server emulates how your entire web page would be on-line.

---

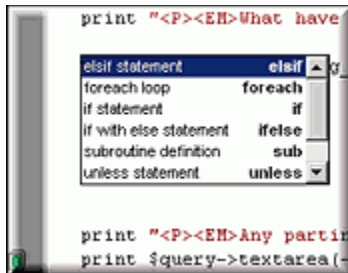
## Live evaluation of expressions



While debugging you can evaluate anything moving...  
As a pop-up hint window or in Watches

---

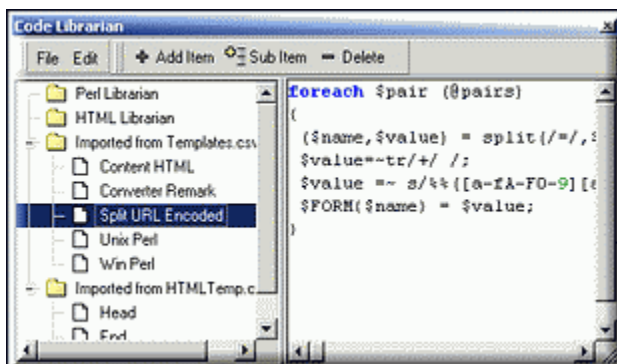
## Rich Editing Features



OptiPerl has syntax highlighting, templates, and many tools for editing

---

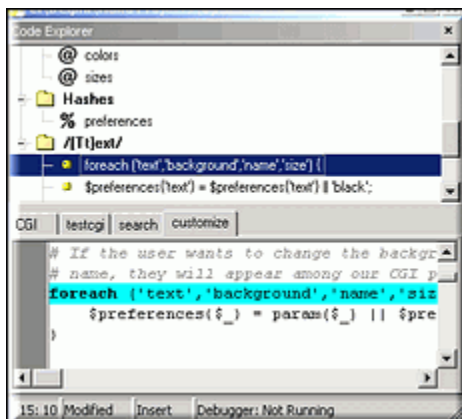
## Code Librarian



Code Librarian stores and shares perl and html code

---

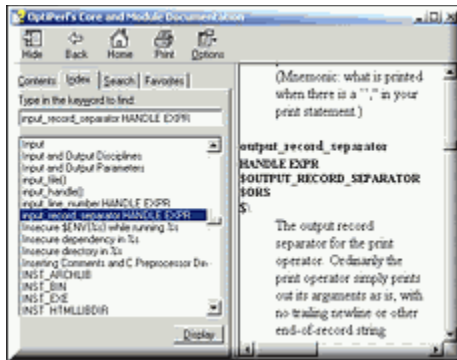
## Navigation in your script



Code explorer lists elements in your program and can perform a perl grep search

---

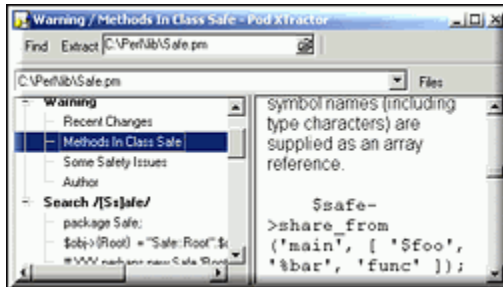
## Context - Sensitive Help



Context-sensitive help with all of perl's core and module documentation and apache documentation

---

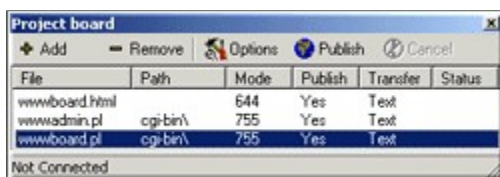
## Pod Support



View POD information, embedded or standalone in a easy to use tree display.

---

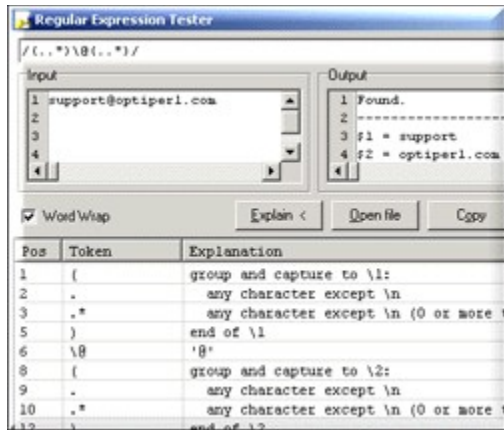
## Projects



Organize many scripts into a project and publish it.

---

## Many Tools



Regular expression tester, perl printer, script uploader, perl reformatter and much much more!

## **Setting Up**

To use OptiPerl you need a distribution of Perl for Windows. We recommend using IndigoPerl Distribution, which is free, easy to install and also contains the apache web server. You can also use the ActiveState distribution.

If you first loaded OptiPerl and you had perl installed, then OptiPerl should have already found it.

If you do not have perl installed yet then:

1) Download perl:

You will find the distribution we recommend on our FAQ page of OptiPerl:

<http://www.xarka.com/optiperl/faq.html>

2) If you downloaded IndigoPerl, unzip it in any directory and run the bat file included. We recommend unzipping the files in `c:\perl`. If you use this folder, then step 3 and 5 below is not needed.

3) Load OptiPerl and open the Tools/Options dialog. Go to tab "Perl" and change to the location of perl.exe.

4) You are ready! If you are new to CGI Perl programming, read more in Basic Stuff.

5) If you have knowledge of apache web server, and want to use, change the settings of External Server in the Server tab. Read more in Using an External Server.



## **Basic stuff**

If you don't know much about how to use CGI scripts in your web page, here is a very brief introduction. There are many great sites on the internet for tutorials.

Make sure you have installed first perl as noted in [Setting Up](#).

Let's assume you want to create a web site with some forms. Do the following

1) Select "New html file" and enter the following.

```
<form action="cgi-bin/basic.cgi" method="GET">
Enter some text here:
<input type="text" name="text" size=30>
<input type="submit"><p></form>
```

This creates an input on your page for the user to enter text. Most Web-Editors allow you to create stuff like this automatically. Notice the GET method used here. When the user presses "Submit", the browser will call the URL:

```
http://127.0.0.1/cgi-bin/test.cgi?text=what_you_typed
```

Under the path you installed OptiPerl, you will find a folder "webroot" (like c:\program files\OptiPerl\webroot). Save your file there with the name "basic.html"

2) Select "New Script" and enter the following.

```
#!/usr/local/bin/perl
print "Content-Type: text/html";
print "$ENV{'QUERY_STRING'}<br>";
```

Save in "webroot\cgi-bin" as basic.cgi.

Now time to run. Make sure that you have a check-mark on "Server/Internal Server Enabled" and the root path under "Server/Internal Server Root" is "c:\program files\OptiPerl\webroot" (or accordingly where you installed OptiPerl).

Also make sure "Server/Run with Server" is checked too.

Go back to your html document "basic.html" pressing the tab in the editor and press "F9" to run. You should see the html document in the web browser. Enter something in the text box and press "Submit"

You should see the result in a dynamically created html page. That's the basics!  
If you get an error, please make sure you read about "[Setting Up](#)".

3) Now to send this to an Internet server, you might need to change the first line (the location of perl on the web server). If your internet web server has cgi enabled with a cgi-bin directory contact your provider for what this line should be.

Also read more in the [FAQ](#).



## **Frequently Asked Questions**

### **General Questions**

*Q: Is OptiPerl a language for making forms on web pages?*

A: OptiPerl is not a language. It is a tool that helps programming in Perl. Perl is a language that makes programs for computers. It just happens that it is also good for making cgi scripts, those used for web pages. Actually almost every time there is some kind of interaction between you and the browser, a perl script is being used, from input boxes to search engines..

*Q: Can OptiPerl help me learn Perl?*

A: It can help you, especially if you want to deepen in CGI programming, because you can do that offline in windows. However you will not be able to magically do stuff, you must still learn Perl.

*Q: Why isn't it easy to make windows run CGI scripts?*

A: Windows was not made to do this. For script to run from the webbrowser offline, a small server must be loaded when we want to in windows, that actually only serves itself, the user of the same computer. OptiPerl does this internally so you don't have to bother with this. However running an external server like Apache is also supported.

*Q: I just started learning perl, I spent some money for a nice book, but I'm not sure yet if I need OptiPerl.*

A: As noted above, OptiPerl can help you learn perl easier and faster. And not only that - you are welcomed to use the unregistered version for 30 days. Even with the limitation it has on the size of the scripts, it can load most examples and small scripts. Whenever you feel you need OptiPerl, go ahead and register.

*Q: OptiPerl can run my script in many different ways - Which is the best?*

A: The best is using the internal server, as you can also load html pages and see how your script works when the pages load it, just like the real thing. Running without a server at all, is good when you just want to run something you just thought of quickly without even saving, or for non-CGI perl. For a complete test of your entire web page, you can use Apache (even though in most cases just having OptiPerl loaded with do as fine!).

*Q: If OptiPerl's internal server is so great, why so much support for running with apache too?*

A: OptiPerl is also used by professionals for web authoring. Using apache is the closest possible thing to a real server that serves the internet (most use apache also). You may be able to test other aspects of web authoring that do not have to do with perl scripts like apache mod's, .htaccess files etc.

*Q: When I press "check for update", I can see there is a new version out. How do I download?*

A: If you are using the trial version, then just download again and install. If you are a registered customer, go to the password protected page sent to you when you registered (you will be notified anyway with an e-mail when a new version is out).

*Q: Is OptiPerl compatible with Windows 2000, NT and XP?*

A: Yes it is. You can also use as an external server the Internet Information Server included with 2000 and XP (read the according section in the help file).

## Working with OptiPerl

*Q: When I load up OptiPerl I see parts of the desktop. Is this normal?*

A: OptiPerl is made using many small windows, not just one big one. So you will see whatever is behind those windows, either it is the desktop or another program. Note that you can drag and drop text into the editor from other programs.

*Q: How do I add items to the Code Explorer?*

A: Code explorer is a navigational tool. It lists elements in your code and is built automatically whenever you edit your script. To go to one of it's elements, double click the line. You can however edit the names of variables in code explorer by left - clicking and pausing on them. Doing so is like performing a search and replace in your script.

*Q: What are those \*.op files saved in the same directory as the script I am editing?*

A: They are files that store items like the to-do list and the queries. If these are not used then they are not written, and even deleted if completely unused (you don't have any todo items or manually added queries). These files are also associated with OptiPerl so if you double-click them in explorer or load them from "Favorites" then OptiPerl will load with the file clicked.

*Q: I closed the editor window and now I can't open it!*

A: Select "Show Editor" from the Editor Menu.

*Q: Can I find an opening/closing bracket in the editor?*

A: Hold down Control while moving the mouse.

*Q: Pressing enter does not move to the next line when typing in text areas in the web browser*

A: Use Ctrl-M instead of enter.

*Q: How can I dock and undock the code explorer window?*

A: Dock it to the main editor window by dragging it. If you want to place it over the main editor without it being docked, hold down the ctrl key while dragging it. Undock it by double clicking on it's title bar.

## Setting Up

*Q: What do I need to run OptiPerl?*

A: You need Perl!. A very good distribution of Perl, that also has Apache with it if you ever need it, is IndigoPerl.

*Q: Do I also need a web server?*

A: No, as OptiPerl can turn on it's internal web server whenever you need it. You can also run scripts without a server at. But if you ever need it, you can use an external server like Apache instead of the internal server.

## Debugger

*Q: While debugging a script that uses CGI module and using a query with a POST method, after tracing the first line, the debugger's status says "Running", and there is no response to other commands.*

A: Just like debugging scripts that have a line that expects user input from <STDIN>, the CGI module does the same to receive POST data. You must enter the data in the "Send" line of the webbrowser to continue.

*Q: Can I load the debugger passing a command line (my script is not a CGI)*

A: Type the command line you want in the query box and select "Command Line" as a "method", before starting the debugger. This way you can receive the command line in @ARGV.

## Running

*Q: When I press run, the first line in the browser is "Content-type: text/html". This shouldn't happen!*

A: You have unchecked "Run with server" in the Server Menu. When this checked, then the script is run through the web server and loaded as a http address. If it isn't then the scripts output is sent to a temporary file, and the file is loaded in the web browser as a file url. Note that this sometimes can be useful. For example, try running a script that creates a cookie without the internal server loaded. The cookie is not created, instead the request that would be sent to the client is shown as the first line. This way you can easily debug cookies. For the cookie to be created of course, load the internal server and select "Run with Server".

*Q: What is "Set Starting Path"?*

A: In special cases you may need to set the starting paths of scripts, mostly for console applications. This selection also affects when running with the console.

*Q: When I run in the console, the screen flashes first.*

A: To fix this, open from the start menu a dos command prompt. You will find it full screen; press alt-tab to make it windowed and then exit.

## Internal Sever

*Q: When I run a script pressing F9, I am prompted to connect to the internet!*

A: If you get this, just press cancel.

*Q: When I load OptiPerl's internal server, I get an error message.*

A: Make sure that in only one instance of OptiPerl you have loaded the internal server. Or select in the option dialog "allow only one instance". Another possibility is that you might have an external server loaded like apache and Optiperl didn't find it. If you use apache, read more in the corresponding section. If you have Windows 2000 or XP, you may have Internet Information Server enabled which causes this problem with the internal server. Again there is more in the rest of the help file.

*Q: Sometimes I get in the status line of the web server "Timeout Occurred", and the output of the webbrowser is wrong.*

A: When this happens, it is because the script entered an endless loop or some error prevents it from exiting, so OptiPerl will try to close it. A script that does this has a bug and is not a correct

program for CGI use. Try debugging it first by disabling "Run with Server" or even running in the console if needed.

One common cause is when the script tries to load an external program that does not exist, causing it to wait endlessly. For example if you use the date command like this:

```
$date_command = "bin\date";  
$date = ` $date_command +"%m/%d/%Y %T %Z" `;
```

In windows, this would cause the above error. You must comment out the second line or use the version converter.

If this is not the case, because your script is just slow and needs a lot of time until it sends out results, then increase the timeout value in the Options Dialog.

*Q: Sometimes the changes in the script I make are not reflected in the web browser. What's wrong?*

A: If you see this happening, then try going to the Control Panel / Internet Options / General tab / Temporary Internet Files / Settings and select "Every visit to the page".

*Q: When I first load OptiPerl, load the internal server and run a script, I have to wait 50-60 seconds until it shows (but afterwards the scripts work fine)!*

A: Go to the Control Panel / Internet Options / Connections tab / Lan Settings button / Automatic configuration and remove the checkmark "Automatically detect settings".

*Q: My command line is not passed to the script.*

A: It doesn't make sense to send a command line to a CGI script. CGI's only accept queries using a GET or POST method.

*Q: When I have the internal server running, I get the output "Forbidden".*

A: Make sure that the script you are trying to run is located under the folder set in the "Internal Server" root path.

*Q: When I try to run a script I am prompted to download!*

A: Make sure that the path to perl is correct in the Options/Perl, and the associations are OK in Options/Internal Server. Press the "Default" button to make sure.

## **Problems**

*Q: When I first load OptiPerl, I get an error message and the program terminates. What's wrong?*

A: You probably have an old version of Windows 95 without Internet Explorer 5 or above. Try installing a newer version of Internet Explorer, or installing Windows 98 and above.

*Q: When I load OptiPerl, I get a Dr. Watson application error (windows NT)*

A: The same applies as above. Windows NT 4 ships with IE 2 and this will not work. You must upgrade to IE 5.

*Q: When I press F1 on a word, OptiPerl exits with an error.*

A: You need to upgrade Windows Html Help to version 1.1 or newer. Get the update at <http://officeupdate.microsoft.com/2000/downloadDetails/Hhupd.htm>

*Q: When I right click on a module in Code Explorer, the module does not open in the editor.*

A: Make sure the %INC path is correct in Options/Perl. Press the default button to make sure.

### **Some free web hosting services with a CGI directory**

Tripod (<http://www.tripod.com/>)

Clan World (<http://www.clanworld.org/>)

Virtual Avenue (<http://www.virtualave.net/>)

World Zone (<http://www.worldzone.net/>)

*Note: If you would like to be included here, please send an e-mail to [optiperl@xarka.com](mailto:optiperl@xarka.com)*

### **Using Apache as an External Web Server (Optional)**

*Q: Why must I enter the exe name of the server program I am using (for example "apache.exe") in the Options Dialog?*

A: From this name, OptiPerl searches often in the memory of the computer to see if the server is loaded. If it senses that it is loaded, then it will turn off it's internal server automatically and also gray out the menu item to load it. You cannot have two servers loaded at the same time. After OptiPerl senses that an external server is loaded, it will use the paths selected in the Options Dialog under external server. Read more in the section "Using an External Server"

*Q: But I haven't loaded apache, and still the internal server option is grayed out.*

A: Probably you installed apache in a windows 2000/Nt system, and apache is running as a service, and not in a dos-box. If you need to turn it off, go to the computer management and press Stop on the apache service.

*Q: How do I load Apache?*

A: Run in from OptiPerl's Tool menu, or select from the start menu the item "Start Apache".

*Q: I get "External Server Loaded" even though apache is not loaded.*

A: If you have NT/2000, apache could be loaded as a service.

*Q: I am getting a 500 error when running.*

A: First make sure your she-bang line (the first line) points correctly to perl. For example, it could be `#!c:\perl\bin\perl.exe`

*Q: When I load Apache, the window opens and then closes immediately, and I don't get a chance to read what it says. What can I do?*

A: Apache must be giving an error message. To be able to see it, load to command prompt to go to dos and load from there. The window will not close, so check what the error message is. Also check the last line of the error log.

*Q: Is Apache loaded if the window opens and then closes immediately?*

A: No it is not. You must always have a console window loaded, that can be minimized, but not closed if you want to run CGI scripts in OptiPerl. This does not apply if you are using Windows 2000 or NT. In this case check if apache is running as a service.

*Q: What is the correct way to close the Apache console window when I've finished using OptiPerl?*

A: An almost-correct way that's fine if you are using Apache just for offline work and not as a web server is selecting the window and pressing Ctrl-C.

*Q: The error message in Apache's log is "failed to get a socket for port 80".*

A: This has to do with the network configuration in windows. Go to Settings/Control Panel/Network, and check what get's loaded there. A good idea is to delete the items there and re-install them, but make sure you know what you are doing. What must be there for sure is a TCP/IP adaptor in Protocols. This applies for both Windows 98 and 2000/NT



## ***Update changes***

List of additions in Optiperl since version 1.

All customers since the first shareware release have received for free the updates, and will continue to get free updates in the future. This is a policy of Xarka Software.

Optiperl is an ongoing project. Each new version comes from implementations of ideas that users of it have and that is why it is the best visual environment for perl.

*We thank you for it's success!*

### **Version 3.5 (March 2002)**

Major upgrade that has the following changes by category:

#### *Code explorer:*

- Rename variables using left-click
- Exports node
- Larger summary in hint window
- Better context handling of variables
- Increased speed of regular expression searching
- Minor bug fixes

#### *Debugger:*

- Ability to add breakpoints in required and used modules of the script, even in run time evaluated modules
- Gutter graphics show valid and invalid breakpoints after debugger has started
- If syntax errors are found then the syntax error checking window is loaded
- Bug with certain versions of Windows NT fixed
- Better "Evaluate Expression" window
- You can change values of variables in the "Watches" window

#### *Editor:*

- Comment out multiple lines
- Synchronized scrolling option when a secondary editor window is open
- Holding down Control while moving mouse over a bracket shows corresponding one
- Typing brackets highlights corresponding one
- While moving in code, the subroutine in view gets highlighted in the code explorer. By selecting Search/Find Subroutine, it will also get focused
- Improved find & replace dialogs
- Fixed problem with international keyboards
- Regular expressions highlighted in color syntax parser
- Increase and decrease indent buttons

#### *Environment:*

- Editing of shortcuts possible
- More options in the Options dialog
- Alt - shortcut keys can be used to access the menu commands and tabs in the editor
- Support for extended keyboard
- Can do a regular expression find on the output and pod viewer browsers
- Minor changes - bug fixes

#### *Running:*

- Better tools support
- Selection in Query Dialog to import the form data of an html page (useful for debugging)
- Selection of starting path of the script possible
- More options when running in console
- Support for third party external browser
- Expanded syntax check evaluates first required modules

#### *Projects:*

- Publishing through firewall possible.
- Internal and external server options saved with each project

#### *Tools:*

- Added File Manager that can do a very fast regular expression search in entire folders
- Improved external tool support
- Backreferences listed for all lines matching pattern in regular expression tester
- View log option in perltidy tool
- Improved perl printer
- Improved pod extractor

#### **Version 3.4 (September 2001)**

- Improved code explorer
- The external programs run by the internal server can be set up
- Pattern search in entire project
- Added "Auto-View" tab in web browser, to view a custom file at real-time.
- Added File Compare tool.
- "Find declaration" added. Holding down Control and clicking the mouse on a declaration will search for it and open it in the browser

#### **Version 3.3 (June 2001)**

- Project support

#### **Version 3.25 (May 2001)**

- Regular Expression tester with "Explain"
- Ability to open many editor windows
- Right click on modules in code explorer
- Monospace font for the editor limitation removed

#### **Version 3.2 (April 2001)**

- Perl tidy tool (source code reformatter)
- Information tab in web browser

#### **Version 3.1 (April 2001)**

- RegExp Tester
- Minor Bugfixes

#### **Version 3.0 (March 2001)**

- OptiPerl is the new name of "Visual Perl Editor"
- Completely rewritten user interface with major improvements.

- Embedded server. Now apache is not required (but well supported)
- Feature packed editor with syntax highlighting
- Improved Code librarian and code templates
- Context sensitive help on core perl and module documentation
- Many new tools
- Improved Script uploader

**Version 2.6 (October 2000)**

- Minor improvements

**Version 2.5 (April 2000)**

- Internal Error testing

**Version 2.0 (February 2000)**

- First shareware release of Visual Perl Editor

**Version 1.0 (December 1999)**

- Visual Perl Editor was released as freeware. It was the first editor for cgi oriented perl using apache server as a base for running perl scripts. The basic design of it (two panels, one to edit the script and another that showed the output as a web page) has been followed ever since.

## ***License Agreement***

### **EVALUATION VERSION**

This is not free software. You are hereby licensed to use this software for evaluation purposes without charge for a period of 30 days.

### **COPYRIGHT RESTRICTION**

Xarka software name and any logo or graphics file that represents our software may not be used in any way to promote products developed with our software. All parts of Xarka's software and products are copyright protected.

### **DISCLAIMER**

THIS SOFTWARE IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL THE AUTHOR BE LIABLE TO YOU FOR ANY DAMAGES, INCLUDING INCIDENTAL OR CONSEQUENTIAL DAMAGES, ARISING OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. YOU ACKNOWLEDGE THAT YOU HAVE READ THIS LICENSE, UNDERSTAND IT AND AGREE TO BE BOUND BY ITS TERMS.

### **DISTRIBUTION**

This software may be distributed freely in its original unmodified and unregistered form. The distribution has to include all files of its original distribution. Distributors may not charge any money for it.

### **OTHER RESTRICTIONS**

You may not modify, reverse engineer, decompile or disassemble this software in any way, including changing or removing any messages or windows.

## ***Usage Guide***

OptiPerl is not a just another tool you have to make perl scripts; It is a complete graphical developing suite for perl.

First make sure you have perl installed. Read more at [Setting Up](#).

Please read the following chapters to learn more about all aspects of developing:

- Creating Scripts
- Editing Scripts
- Testing for Errors
- Debugging
- Running
- About the internal server, using external server and file handling
- More tools at your disposal

Having problems? Might get an answer in the [F.A.Q.](#)

Our support is also here to help you! E-mail us at [optiperl@xarka.com](mailto:optiperl@xarka.com).

## ***Creating Scripts***

OptiPerl can help you make and test perl scripts that are

- Console
- CGI that are executed directly from a URL, for example `http://www.myweb.com/cgi-bin/run.cgi?query`
- CGI that are run through a form's GET or POST method.
- Server-side includes.

If you want to create a new script, select *File/New Script*, or load an existing one with *Open*. If the script you are about to edit is associated with a html document that has forms or uses the CGI, you may also want to load that html document or create a new one with *File/New Html*.

Also saving in Unix or Windows format is supported. Select the line endings in the File Menu

Included with OptiPerl at location {Install dir}\webroot is the example "test.html" and in the cgi-bin folder the script "test-form.cgi" that the first uses.

## ***Editing Scripts***

The main editor of OptiPerl is a sophisticated and highly customizable editor. Here is a list of features of the editor:

- Syntax highlighting for both perl and html code. The colors may be customized in the Options Dialog.
- Templates that are invoked with CTRL-J.
- Bookmarks, accessed by right clicking on the editor.
- Find/Replace
- Version Converter
- Context-sensitive help with F1.
- Bracket matching by holding down Ctrl and moving the mouse, or pressing Ctrl - \
- Holding down Control and clicking the mouse on a declaration will search for it and open it in the editor.

Also linked to the editor are many tools for building perl scripts.

Depending on what kind of document you are editing, perl or html, accordingly is the syntax highlighting.

## ***Testing for Errors***

In OptiPerl you can test your script for compilation errors using two methods by selecting *Run / Syntax check* or *Expanded Syntax Check*. Both will check your script for errors or warnings and report the results.

- Syntax check outputs results identical to running perl with a -c switch.
- Expanded syntax check will also evaluate first required modules in your script. This is useful if you are using required modules that declare a variable, and in the script you get false warnings of "used only once".

The Status Window is brought up docked to the main editor. By double clicking each line, you can go to the offending line. If you used the expanded syntax check, you might also get errors and warnings in required modules, which optiperl will load first before going to the line.

In the Options under "Perl" you can select the level of warnings (None,Useful,All).



## **Debugging**

OptiPerl enables you to trace through your script, watching the output of variables after each line and evaluating expressions.

Most of the commands for debugging are in the [debugger menu](#).

To start the debugger, select "Start Debugger"

After it is loaded, the line which is about to be run is focused in the editor.

At this point, you can:

- Add breakpoints to your script, by clicking the gutter next to the line you want to add the breakpoint. Breakpoints are selected lines that when encountered in the programs flow, execution is stopped. After the debugger loads, valid breakpoints are noted with a tick graphic in the gutter, while invalid have an x mark.
- Open the watch window. In the watch window you may enter expressions that are evaluated throughout the programs flow. To enter a watch, right click the window and enter the expression. You can also change a value of an expression, by typing it in the "value" field. For example if your programs uses the a hash named %h you may enter "%h" to see how it changes while your program is working. Actually you may enter any expression, not only variables.
- If Auto Evaluation is Enabled, then hover the mouse over any keyword on the editor, to see it evaluated in a hint box. The results from this is identical to adding an expression in the watch window, however a lot easier to use. Note that the word under the cursor is evaluated OR the selected text under the cursor. Selecting first the text can help you quickly evaluating expressions with spaces like

```
print "<b>$k</b> = ".$ENV{$k}."<p>";
```

Select first the above expression and the hover the mouse on it.

The programs flow is control from the "Tracing" commands. These are

- Single Step: Runs one line, entering subroutines or libraries if needed.
- Step Over: Run one line, stepping over subroutines.
- Return from subroutine: If pressed in a subroutine, will run until the subroutine exits. If not run in a subroutine, will run the script until it finishes, or until a breakpoint is found.
- Continue script: Continues running the script until it exits or a subroutine is found.

Often while debugging, you may enter code that is in a different file. If so that file will automatically be loaded in the editor.

The debuggers status is displayed in the status line of the main editor. Status can have the following values:

- Not loaded: The debugger is not loaded. Press "Start Debugger" to load.
- Stopped: The debugger has stopped on a line.
- Processing: The debugger is processing a command you have sent it.
- Terminated: The debugger has finished execution of the script.

- Running: The script is running.

After your script terminates, you have an option to "Stop Debugger" or "Restart Debugger". If you will not be re-debugging your program, "Stop" is recommended, so that the debugger will be unloaded and the resources freed.

While running the script, all output is sent to the Web Browser, so you can view the output in either text or html format, by selecting the corresponding tab.

Your script might also read from <STDIN> (if for example it expects user input from the keyboard, or a post method). If this happens, you will see that the status will be "Running", as the program is expecting user input. Enter the input in the edit box of the Web Browser and press "Send" to continue.

While running the script or ever after it has terminated (but with the debugger running) you may also execute the following:

- List subroutine names
- List variables in package
- Methods callable

The above can be executed with entering a perl search pattern first or without

- Evaluate expression: The result from this is identical to an expression entered in the watch window.

Note: To load the debugger while sending the script a dos command line, select as a "method" in the query editor the command line. This helps in debugging perl scripts that are not ment to be CGI's.

## ***Running Scripts***

As perl scripts can be used for many purposes like for standalone programs or CGI scripts, OptiPerl can run scripts or html pages with scripts in many ways depending on what you need.

Scripts can be run in

- The internal web browser. Running them here has the advantage of viewing the output in both text and html format.
- An external web browser. This launches the script or the html document that uses the script in the default browser.
- In a dos console. This launches the script in a dos window using whatever parameters you want.

When viewing the scripts output in the web browser, you can click the tabs in the window to select a text or web view.

The above also apply when running (or better loading) an html document in the editor.

If you want to run the script using the internal or an external server, it is also important where you save your script/html page.

Read more about running at:

- [Running with the Internal Server](#)
- [Running without a server](#)
- [Using an External Server](#)

## ***The Internal Server***

OptiPerl can help you test not only a perl script itself, but an entire web page with many html documents and scripts.

Usually you cannot test a perl cgi script by running it directly, because it does not make any sense to run it this way. Most script have some kind of input from html documents on the internet, so you can't test it if you don't upload it to the server.

However OptiPerl makes it easy if needed to run the scripts through a server, emulating a distant server. When the Internal Server is enabled (selectable at the "Server" Menu) the following happen:

- Your computer becomes a server and a client at the same time, even if you are offline.
- The IP address your computer listens to become "http://127.0.0.1" or "http://localhost"
- Web server logs are created in the [Log Windows](#).
- Any file or CGI script you have under the folder selected by "Internal Server - Root" is accessible by the http protocol. For example:

Try selecting from the Server Menu a server root folder of

```
c:\program files\OptiPerl\webroot
```

and make sure you have "Internal Server" enabled (checked). Also make sure you have "Run With Server" checked.

Included with OptiPerl at the above folder is

```
test.htm and  
\cgi-bin\test-form.cgi
```

So boot up your favorite browser and go to address

```
http://127.0.0.1/test.htm
```

The test document is loaded. Now enter some input in the form fields and press "Submit". With "Submit" the test-form.cgi script is loaded from the server and you can see it's output.

So OptiPerl created a mini-server on your computer!

Read more on [Basics](#) if you are new to cgi scripts to get see how the two files worked together.

If you are more experienced, you may notice that the "shebang" line (the first line with #! that loads perl) can point to anything, and does not have to point to a real file. This is useful to send the scripts later to a real internet server without changing the path to perl.

Of course most developing happens in OptiPerl. So without changing the above settings, open test.htm and test-form.cgi. Select test.htm and press F9 to run it. You could also select test-form.cgi and enter a query in the [query editor](#).

So if you are going to develop scripts that you will test using the internal server, it is important where you save your files.

This is actually good news! Let's assume you have a web with many folders and html documents in the folder "c:\my web\bestweb". You want to enhance your web adding cgi-scripts. So just like most internet servers, create a cgi-bin folder at "c:\my web\bestweb\cgi-bin".

And put there all your cgi's. These cgi's are referenced from your pages as  
cgi-bin/testcgi.cgi

So for example in a form's action you would put  
<form method="GET" action="cgi-bin/test-form.cgi">  
(or method="POST")

Now just select as the "internal server root" in OptiPerl the folder  
c:\my web\bestweb

You are ready! Load your index.html in OptiPerl' s internal browser, and you can check your entire web page with all it's scripts working. When everything is OK upload your web, preserving the folder structure to the server.

The internal server let's you also set up what external programs are used for each type of file. These associations are set-up under the internal server tab in the Options dialog. You probably won't need to change these, the default settings will probably be OK (press the default button under the box to get the default settings).

Also read the section "Internal Server" in the [FAQ](#)

You can also upload CGI script as you make them using "[Script Uploader](#)".

## ***Running without a server***

When you have "Run with Server" enabled, the scripts you run are sent to the internal web browser as a http address, for example

```
http://127.0.0.1/cgi-bin/test-form.cgi
```

However, you can also test scripts without using the internal server, by deselecting "Run with server". This way the path to the file is sent to the browser instead, for example:

```
"c:\program files\OptiPerl\  
webroot\cgi-bin\test-form.cgi"
```

If you use this method, you cannot load a html document that sends it's output to a cgi script. You can however send directly a query, with either GET or POST method.

Another option is sending a command line to the script (useful for perl code that's not meant for CGI). Select the Command Line "method" in the query editor. Note that selecting this when the internal server is loaded does not do anything, as it doesn't make sense.

## ***Using an External Server***

You can also run your scripts by utilizing a third-party server program, however you must have some experience to set up the server program.

Here we will explain how to set up Apache to run your scripts off-line. This section is not necessary, but you might want also to test your webpage under a real server. This might especially interest you if your internet provider uses Apache as it's server. Most internet hosting services do use Apache.

If you are interested in using Internet Information Server, then read the corresponding section of the help file.

If you have installed IndigoPerl as instructed in "Setting Up", then you already have apache installed. This is because IndigoPerl distribution of Perl also has Apache included, exactly so you can test your scripts offline. It has also set up the httpd.conf file of Apache so you do not have to change it.

What you must do is set up the folders in files in Options Dialog/Server:

The below defaults apply if you installed IndigoPerl in `c:\perl\`

- Access log file: `c:\perl\logs\access.log`
- Error log file: `c:\perl\logs\error.log`
- Executable: `apache.exe`
- CGI Folder: `c:\perl`
- HTML Folder: `c:\perl\htdocs`

Executable is the exe name of the server of you are using, not the full path to the server. Giving the correct executable name, enables OptiPerl to know when the server is loaded or not and if so send the correct relative directories to the URL when pressing "Run". If you load or unload the external server while OptiPerl is running, you can check if it has found it in the Server Menu (You will see a label "External Server Loaded")

If you are using Windows NT and OptiPerl still cannot determine if the server is loaded or not, then manually press "Consider External Server Loaded".

The log files are used to have a live view of them in the Log Windows

**Note:** Notice that the "CGI Folder" is `c:\perl` and not `c:\perl\cgi-bin`, while the HTML Folder is `c:\perl\htdocs`. This is because of the relative position of files that must be maintained. For example:

- `http://127.0.0.1/index.html` points to `c:\perl\htdocs\index.html`
- `http://127.0.0.1/folder/index.html` points to `c:\perl\htdocs\folder\index.html`
- `http://127.0.0.1/cgi-bin/test.cgi` points to `c:\perl\cgi-bin\test.cgi`

Apache runs by selecting "Start Apache" from the start menu. Start it (a console window - minimize it) and your web browser, and enter `http://127.0.0.1` (you don't have to be on the internet). If you get a message "Go offline" press cancel. You should see the Apache's help. If

not, try putting any html document named index.html in the directory c:\perl\htdocs, or c:\apache\htdocs.

Note that whenever CGI Scripts are to be run, apache must be loaded and working. If apache's window opens and closes very fast, it has not been loaded. Check the [F.A.Q](#) if this happens.

Whatever CGI script you put in directory C:\perl\cgi-bin, can be run by putting an address in your browser <http://127.0.0.1/cgi-bin/test.cgi> etc. A good idea is to try this also. Copy hello.cgi into the directory c:\perl\cgi-bin and enter in your browser the above address. If you get errors, something is wrong.

6) **IMPORTANT:** *To run CGI scripts in windows using Apache, you must change their first line to*

`#!C:\perl\bin\perl.exe` *(where you have your perl interpreter)*

*When you upload them afterwards to the server, change to*

`#!/usr/local/bin/perl` *(or wherever perl is at the server)*

Optiperl can change the above automatically when uploading either when publishing your [project](#) or using the [script uploader](#).

Note that the above is only needed when you use apache. If you are using the internal server you can put anything there when testing locally.

Also read the section "External Server" in the [FAQ](#)



## **Using Internet Information Server**

Internet Information Server (IIS) can be used instead of OptiPerl's internal server or apache in a windows 2000 or NT environment. To use it with OptiPerl, you need ActiveState's ActivePerl installed also.

Windows 2000 Professional ships with IIS. To install go to Control Panel / Add remove programs / Add Windows Components and check the checkbox "Internet Information Services". Install all of it's components.

After installing, find the folder `c:\inetpub\wwwroot` and create in it the folder `cgi-bin`.

Now go to Control Panel / Administrative Tools / Internet Services Manager and note the name of your computer. Expand this node and then expand the node "Default Web Site". You will see as a node the newly created folder `cgi-bin`. Select it and press "Properties".

On the tab "Virtual Directory" select "Execute Permissions" as "Scripts and Executables". Also press the button "Configuration" and then the button "Add". In the "Add/Edit Application Extension Mapping" window, enter the following:

Executable: `C:\Perl\bin\PerlIS.dll` (make sure you have ActiveState's ActivePerl installed)  
Extension: `pl`

Repeat the above for the extension `cgi` and press ok to save settings.

Go to Control Panel / Administrative Tools / Personal Web Manager and press the button "Start" if the service has not already started. Press the link in the window that starts with `http://` to test if it is running (e.g. `http://computer`)

To test that everything is OK, copy all the examples of OptiPerl from `\program files\optiperl\webroot\cgi-bin` to the folder `\inetpub\wwwroot\cgi-bin`. The scripts should be accessible by entering in your browser `http://computer/cgi-bin/script.cgi` (replace `computer` with the name of your computer shown in the window).

To set up OptiPerl, go to Tools / Options / Server and enter in the "External Server" box:

External Server: `inetinfo.exe`  
CGI Folder: `c:\inetpub`  
HTML Folder: `c:\inetpub\wwwroot`  
and press OK.

Load now a cgi script from the folder `c:\inetpub\cgi-bin` and run it. Make sure you have selected the option server /run with server.

All scripts you create must be saved in the above folder. Note that if you have enabled IIS, then the internal server of Optiperl will not work. You can easily start and stop IIS from the window Control Panel / Administrative Tools / Personal Web Manager.

Using IIS is recommended if you have a web page that uses FrontPage extensions and want to test it offline.



## ***Running in Console***

Running in console is useful mostly to test non CGI perl code. When selecting the item from the menu, a parameter dialog is brought up that prompts you for parameters to run with.

You can also select a starting path for console script in the "Run" menu.

The default parameter (and the simplest and necessary for this to work) is "%pathsn% %query %". This gets replaced with the scripts path as a short name, and also adds the line you have in the query box. So if you have a script named `c:\perl\test.pl` then the above would run perl in a console like this:

```
perl c:\perl\test.pl
```

Running with a parameter of `-d "%pathsn%"` would run perl invoking perl5db for example

```
perl -d c:\perl\test.pl
```

You don't have to put %pathsn% at all. You can replace with any parameter you want, even create one-liner programs extremely quickly. For example try

```
-e "print 'Hello';"
```

Note that the command line options entered here are not affected from the query editor, even if you have selected the command line as a "method".

If you are running in the console and the output is large, you might need to pause on each page. To do this, after pressing "Run in Console", enter in the dialog box:

```
%pathsn% | more
```

This way after each page of output you will be prompted to press a key for the next page.

## ***Working with Projects***

Often for a complete web page, two or more scripts are used, that might also share a common library. For example a script that produces an on-line poll might also have another script for the administrator of the poll.

A project in Optiperl is a set of perl scripts and/or html pages that all belong to the same web site. It is useful to organize many such scripts and html pages if needed in a single project so you can:

- Load the project in the project window and quickly access the script you want to edit,
- Publish all modified scripts to your web server,
- Save properties that have to do with all the scripts in the project.

To create a project select from the project menu the item "New". You can add files to the project by selecting "Add" that will add the script that you are currently editing, or "Add Files" to see an open files dialog to add manual one or more files.

Each file is listed in the Project Window. Double clicking a file will open it in the editor. Right-clicking will bring up a pop-up menu to edit the properties for each field.

The project window consists of 6 fields:

- File: The filename of an item in the project
- Path: The relative path of where the file
- Mode: The unix permission that will apply when you publish the file.
- Publish: Whether the file will be uploaded to your server.
- Transfer: Transfer mode for the file. For perl scripts and html pages this is "Text"
- Status: After publishing, this field will show the status.

You can add in your project files that are even in different directory levels. When publishing, the directory structure is recreated if needed on your web page to match the way you have it off-line.

**Important:** You must save your project in the same folder where you have your scripts stored. If you have scripts in many directories, store it in a folder above the directory tree. This is usually the folder "cgi-bin".

See also "[Project Options](#)"

## **Common Tasks**

This document is a guide for the following:

- Using SSI (Server Side Includes)
- Using m4 macro processor
- Using pod2html
- Pausing for each page in console output

### **Using SSI (Server Side Includes)**

The internal server of OptiPerl does not support SSI. However you can use apache as an external server for offline testing.

First create in the \htdocs folder a file named ".htaccess" (notice the starting dot) with the following text:

```
AddHandler server-parsed .shtml
Options +Includes
```

Now all html files added in this folder that have an shtml extension will be SSI enabled. For example:

- 1) Copy the "hello.cgi" script into the \cgi-bin folder of apache
- 2) create a file named "hello.shtml" in the \htdocs folder with the following text:

```
<html>
Running an SSI<p>
<!--#exec cgi="/cgi-bin/hello.cgi" -->
</html>
```

You should see the following output when running the shtml file (make sure apache is running)

```
Running an SSI
Welcome to OptiPerl!
```

### **Using m4 macro processor**

You can add a command in the Tools menu to process a script using GNU's m4 macro processor (available at <http://www.gnu.org/software/m4/m4.html>). Using "configure tools", add the following:

```
Program: c:\m4\bin\m4.exe
Parameters: -P "%Path" > "%Query"
```

This will process the file open in the editor and send all the output into filename inputed in a dialog.

### **Using pod2html**

Program: c:\perl\bin\pod2html.bat

Parameters: -infile="%Path" -outfile="%PathNoExt".html

This will create a filename same with the pod file open in the editor with an html extension containing the html documentation

### ***Pausing for each page in console output***

If you are running in the console and the output is large, you might need to pause on each page. To do this, after pressing "Run in Console", enter in the dialog box:

```
%PathSN% | more
```

This way after each page of output you will be prompted to press a key for the next page.

## Version Converter

The version converter is for quickly making two or more versions of your script because of the changes that have to be made in the local version of your script (the one used when off-line for testing) and the one you want to publish. Usually changes have to be made that include:

- Some commands that do not work under windows like `flock`
- Paths to files that Perl writes that are Unix specific. For example if you want to write a file like `"/docs/logs/count.log"`, in your local version in windows, this must be changed to `"count.log"` or `"\cgi-bin\docs\logs\count.log"` if you have created those two directories under the server's root.

To fix these problems, you can modify your scripts code to run in two ways

First add somewhere in the beginning of the script this line:

```
#@Local#?#@Server if you are editing the local version or
```

```
#@Server#?#@Local if you are editing the server version.
```

This way OptiPerl "knows" what version is run. After adding the above, in the search menu you will see which version you are editing in the "Version: xxx" line.

### Example

Enter the following lines:

```
#@Local#?#@Server
#?flock(OUTF,2);
my $semaphore_file=
  'counter.sem';#?'/tmp/counter.sem';
```

Notice that in the Search menu you will get a label "Version: Local"  
Now press "Swap Version", and the above will change to:

```
#@Server#?#@Local
flock(OUTF,2);#?
my $semaphore_file=
  '/tmp/counter.sem';#?'counter.sem';
```

Notice that all lines with a '#?' where swapped at that point.  
Also the search menu will be updated with the label "Version: Server"  
If you press again "Swap" then the first version will be back again.

You can also add longer lines with many version remarks "#?", for example:

```
#@Version A#?#@Version B#?#@Version C

print "A";#?print "B";#?print "C";
```

Here each time "Swap" is pressed, the next version is selected. Accordingly the Version is

update.

If you do not add the `#@Server#?#@Local` line, then the version converter will still work, but OptiPerl will not know which version you are editing.

There is an advantage to letting OptiPerl knowing which version you are editing and using specifically the labels "Server" and "Local": It will warn you if you try to upload the file with FTP and "Local" is selected. Also in the project manager, selecting the option "Convert to Server" will make the conversion if necessary automatically and in the background before uploading.

Included is `counter.cgi`, that demonstrates writing two versions of a script.



## ***Tools for use***

OptiPerl has a plethora of tools to help you write scripts. Here is a synopsis of them and where to read more:

- [Code Explorer](#). Tree view of your scripts elements
- [Code Librarian](#). Library to store perl and html code snippets. Also enables you to share them.
- [Compilation Status](#): Tests your script for warnings and errors.
- [External Tools](#). Add external programs to the [Tools Menu](#) and associate them with your script.
- [Editor Templates](#). Small code templates that can be added with keywords in your program.
- [Log Window](#). Show error and access log items of the internal or external server.
- [Main Editor](#). Read more about the main editor of OptiPerl.
- [Options](#). Customization of OptiPerl.
- [Perl Documentation](#). Provides easy searching for keywords in the main editor.
- [Perl Information](#). Read version information about the build of Perl you are using.
- [Perl Printer](#). Convert lines of text to regular print commands for perl.
- [RegExp Tester](#). Quickly test regular expression.
- [Pod Extractor](#). Extract and view pod info from files.
- [Perl Tidy](#). Reformats your source code.
- [Query Editor](#). Edit the query that is sent to CGI scripts.
- [Script Uploader](#). Upload your script to an FTP server.
- [To-Do List](#). To-Do items associated with the script you are editing.
- [URL Encoder](#). URL Encode and Decode strings.
- [Watches Window](#). Evaluate selected expression during the flow of your program.
- [Web Browser](#). Shows the output of your script as text or web document.
- [Projects](#): Organize many scripts in a single project and publish them.
- [File Explorer](#): Views files and folders in a tree-like view and can do a regular expression search in entire folders.

## File Menu

The file menu has standard commands for file handling, printing and exporting.



When you create a New Script, a filename is given, and if run or debugged, it is first saved temporarily without prompting in the "Default folder for Scripts" found in the [Options Dialog](#).

Accordingly when you create a new html file, it is saved in Default folder for Html files.

Note that if you are to run a newly created script through the [internal server](#), make sure the root folder of it points to the correct folder.

For example, if you have a default folder for scripts as `c:\webs\cgi-bin`, and a default folder for html as `c:\webs`, then if you run the script through the internal server, it's root must be `c:\webs`. This is a limitation if you are using a server, if you run directly with "Run with Server" off then it will work anytime.

About creating html files, you do not have to use OptiPerl to edit and create them, you can use your favorite html editor instead. This is more useful if you need to create something for a quick test, or to create a temporary file that you will use to copy html code into the perl script, with print statements (see also [Perl Printer](#)).

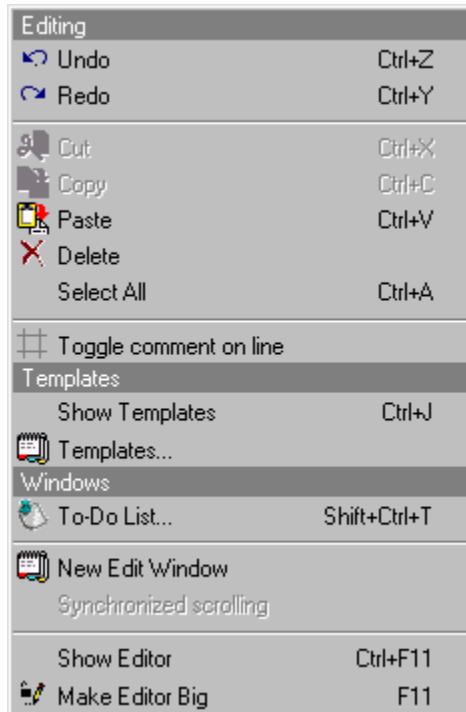
Another use of editing html files, is to fine tune html files that where already created with your favorite editor.

In the [Options](#) Dialog, you can reset the recent files list, or set the maximum number of files they can hold.



## Edit Menu

The edit menu holds commands for standard editing plus other for helping you write your script.



Pressing F11 or F12 will quickly make the editor window or browser window big.

The option New Edit Window opens the selected file in the editor in a secondary window. If you also select the synchronized scrolling, then moving the main editor will also scroll the secondary open windows, which is useful for comparing files.

Also here you can edit the [templates](#) and access the [To-Do list](#).

## Search Menu

The search menu has commands for standard find and replace, and also commands for searching with a Perl Pattern and the Version Converter.

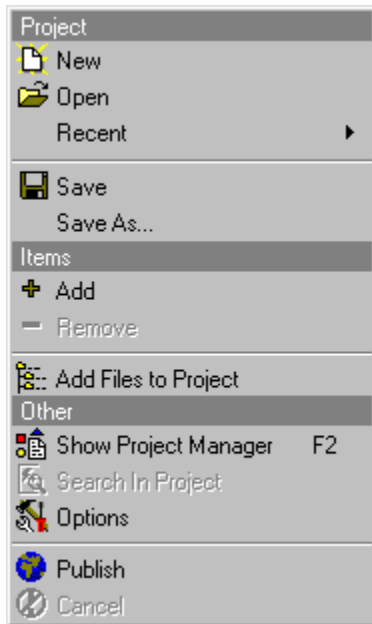


The results of the perl pattern search can be viewed in [Code Explorer](#). Also read about the [Version Converter](#).

Selecting "Find Matching Bracket" will move the cursor to the next bracket. If pressed while the cursor is on a bracket, then it will move to the corresponding one.

## ***Project Menu***















The project menu is used for handling active projects.



Search In Project is identical to "Perl Pattern Search" however will search in all text of the project.

## Debug Menu

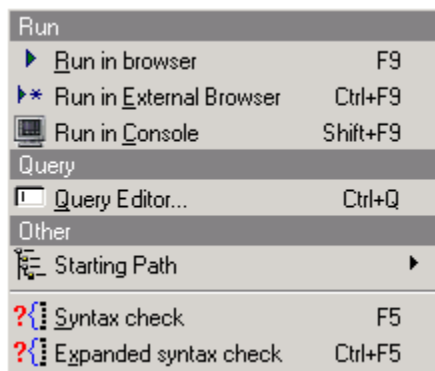
With the debug menu you can debug perl scripts.

Debugger Status		
 Start debugger		F6
 Stop Debugger		Ctrl+F6
 Restart Debugger		Shift+F6
Program Flow		
 Single Step		F7
 Step Over		F8
 Return from Subroutine		Shift+F4
 Continue Script		F4
Script Information		
 List Subroutine Names		Shift+Ctrl+F4
 List variables in package		Shift+Ctrl+F5
 Methods Callable		Shift+Ctrl+F6
 Evaluate Expression		Shift+Ctrl+F7
Tools		
 Add/Remove Breakpoint		Ctrl+F8
 Watches...		Ctrl+F7
 Live Evaluation Enabled		Shift+Ctrl+V

Most of the command here are explained in "[Debugging Scripts](#)". Also take note of the useful "Script information" commands. These can be useful for editing scripts. Also you can select from here if live evaluation will be on when debugging.

## Run Menu

Run menu has commands for running a script or html page, and also test for compilation errors.

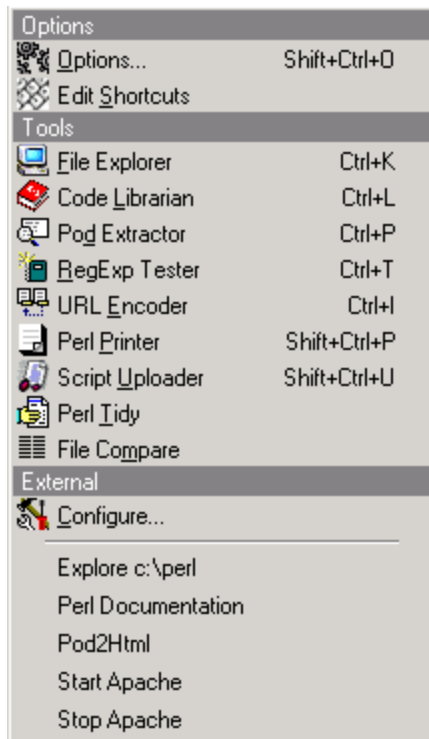


OptiPerl supports running scripts in many different ways. Read more at [Running Scripts](#). Also pressing F5 at any time will test for compilation errors, and report errors or warnings. You can also select a [Starting Path](#).



## Tools Menu

The tools menu has a plethora of internal tools you can use, and also lists any external programs you have configured.



From here you can also access the [Options Dialog](#). Here is the list of internal tools:

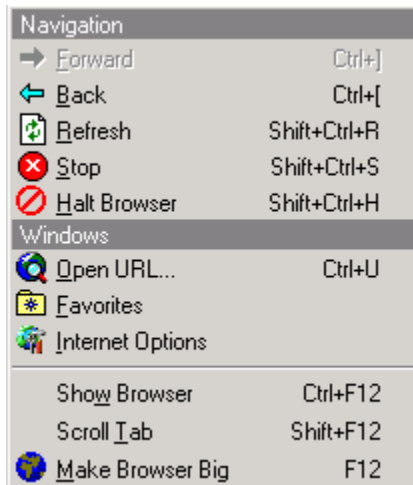
- [Code Librarian](#)
- [To-Do List](#)
- [Pod Extractor](#)
- [URL Encoder](#)
- [Perl Printer](#)
- [RegExp Tester](#)
- [FTP Uploader](#)
- [Perltidy](#)
- [File Explorer](#)
- [File Compare](#)

You can also configure external programs from the "[Configure Tools](#)" Dialog.

Selecting "Edit shortcuts" brings up a dialog for you to customize all shortcuts in OptiPerl.

## Browser Menu

The browser menu has a standard set of browser navigation commands.



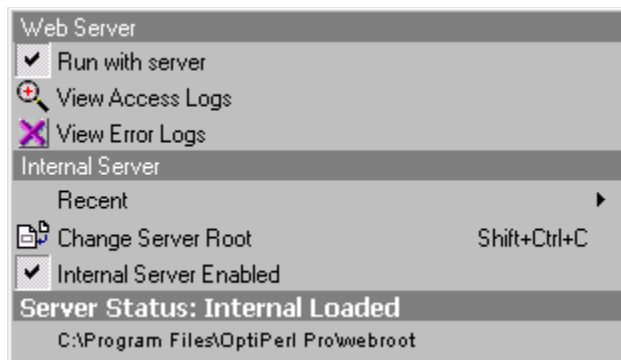
A special command is "Halt Browser". This is like a more powerful "Stop", that can be used to stop a script that was ran through the internal or external server. It is useful while debugging scripts that by mistake have entered an endless loop. This command will stop loading the script and also delete from memory the instance of perl that was running the script.

Note that this is not useful when running the script without a browser. If this is the case, you will be prompted to terminate the script after the allowed timeout set in the [options dialog](#).

You can actually use OptiPerl's browser to browser the internet. Start from a URL in "Open URL". You have the added advantage that all html output can instantly be viewed as text if you select the "Text" tab in the [Web Browser Window](#).

## Server Menu

With the Server menu you can control the internal browser, or see the status of any server loaded.



Read more about these important features of OptiPerl in

- [Running Scripts](#)
- [Internal Server](#)
- [Running without a Server](#)
- [Using an External Server](#)

The last line shows what kind of browser is loaded if any, and shows its root folder. If the internal server is loaded, then the root shown is selected from "Change Server Root". If the external server is loaded, then the folder is the CGI path of the external browser, selected in the [Options Dialog](#).

You can also view the access and error [logs](#) of either the internal or external browser.

## ***Windows Menu***

In the windows menu you have a complete list of all open windows. Selecting a title will bring up the corresponding window if hidden. This same menu can be accessed from the tray icon of OptiPerl.

## ***Custom Skins***

You can add your own background in the menus, editor and code explorer. Just convert your favorites pictures to windows bitmap format, rename them as following and place them in the path where you installed optiperl (for example c:\program files\optiperl).

- OptiEditor.bmp for the editor.
- OptiExplorerer.bmp for the code explorer.
- OptiMain.bmp for the menus.

This works only in the registered versions.

### ***Add server associations***

The server associations affect the internal server. This list contains associations of file extensions and external programs. The default is the association of cgi,pl,plx files to perl.

Practically this means that when the internal server encounters a file with an extension of cgi for example, it will load perl to run it, feed it with data of a POST method if needed and capture the output to display in the web browser.

## ***Check for Update***

Here you can check if a new version of OptiPerl has been released.

If you are a registered customer, you are entitled to the registered new version, so login to your update page and download it. If you have lost your password, then go to the "Customers" section of Xarka Software's homepage.

## ***Code Explorer***

In the code explorer window, you have a centralized view of the subroutines, variables and libraries used in your script. Also the results of Perl Pattern Search are viewed here.

By expanding the tree nodes you can view the parts of your code. Double clicking an item moves the cursor to that item.

If viewing a module under Uses or Requires, you can also right click with the mouse to open the module in the editor. Make sure that the %INC path in Options is correct for this to work.

If had previously done a Perl Pattern Search, then the results of the search will be shown here, under the node "/pattern/", where pattern is what you had searched for.

This window can be docked to the main editor window by dragging it. If you want to place it over the main editor without it being docked, hold down the ctrl key while dragging it. Undock it by double clicking on it's title bar.

You can also rename variables using code explorer. Left click on a variable and pause until you can edit it. This is like doing a search and replace in your script.



## ***Code Librarian***

With the code librarian you can store snippets of perl and html code, for use in your programs. You can:

- Organize the snippets of code using categories and sub categories utilizing the tree-like view.
- You can also share code by importing and exporting the library.

To add a new item, press "Add Item". A new item will be inserted below the focused item with the title "new snippet". You can change it's name by clicking the name once with the mouse. "Add sub item" works like above, but creates a new tree-level under the focused item. Using then "Add Item" you can enter code snippets under the new category.

You can share your code by saving your library. Select "Export Library" to save the snippets to a file. The files you create can then be imported using "Import Library". When importing a library, two new nodes are created, with the html and perl main nodes following the filename of the imported file.

Drag and drop is also enabled. To reorganize you collection, drag an item holding down the left mouse button and releasing it where you want.

"Import CSV" enables you to import a text file, comma delimited, with the format "name","code" for each line in the file and using "%c%n" for the line changes within the code itself. This was the format used in Visual Perl Editor 2.x

## ***Compilation Status***

This window is popped up when you hit "Test for Errors", showing errors and warnings in your script. Double click a line to go to the offending line.

The level of warnings reported is selected in the Options Dialog.

## **Configure Tools**

You can add your own external programs under the tools menu, and associate them with the script you are editing.

In this form you can set up these programs.

To add items, use the following fields:

- Name is the display name in the Tools Menu
- Program is the external program you want to run
- Parameters are the command line parameters for the external program.

In the Parameters field, you can add some tags to associate the script you are editing. Right click and select the "Build parameters" item in the popup menu to invoke a [parameters](#) dialog.

### Example:

#### Using pod2html:

1. Enter pod2html as name
2. Enter "c:\perl\bin\pod2html.bat" as program.
3. Enter the following as parameter:

```
-infile=%PathSn% -outfile="%folder%\%FileNoExt%.html"
```

Notice the use of double quotes to make sure that paths with spaces are not interpreted from windows as multiple parameters.

4. The above will extract the pod text in your script in an html file. The file will be created in the same folder.

You could also set the above up so that the name of the output file is queried first:

```
-infile=%Pathsn% -outfile="%Query%"
```

## ***Editor Templates***

Editor templates are invoked by pressing CTRL-J in the editor. When pressing the key sequence, a small window is brought up with common routines that are used in perl. Selecting one will insert the text in your editor. Alternatively you can enter the name of the code template in the editor, and press CTRL-J on it. This way the name will be replaced with the code, without showing the window.

You can also edit the templates and add your own. Select "Edit/Templates...", to open the corresponding window. Here you can add, edit, sort and delete templates. To add a template, do the following:

- 1) Press Add Template
- 2) Go over the text of the newly inserted line "New Template" and press once the mouse button to edit the name of the template
- 3) Repeat with "Enter Description" to add a description
- 4) Edit the code in the text editor.
- 5) Optionally sort the templates.

In the text editor, enter the character "|" to show where you want the cursor positioned when you insert the template.

You can also import templates from a txt file.

## ***File Compare***

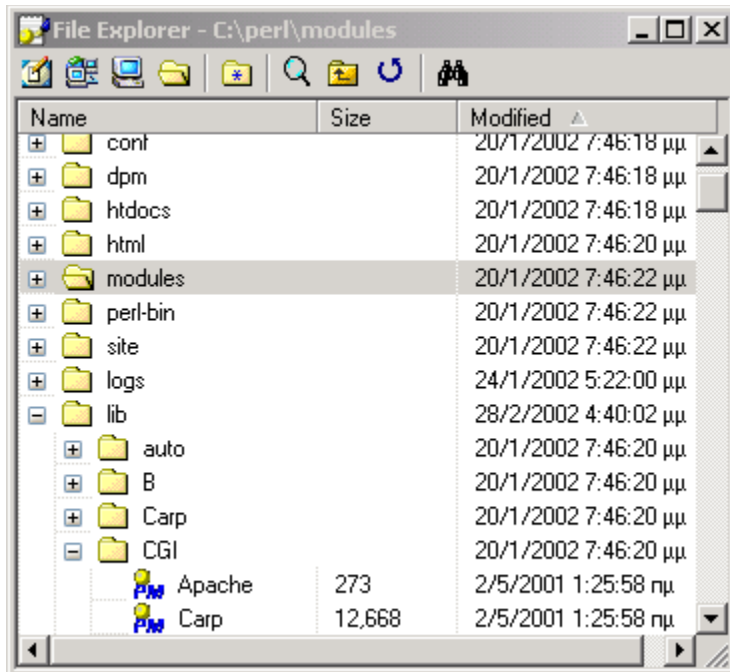
The file compare tools is handy to view two file's differences. Press the buttons to select the files you want and then the compare button to start.

## ***File Explorer***

The file explorer can be used to browse a folder and open files. Right clicking on a folder or file will open an explorer menu.

Double-clicking a file associated with OptiPerl will open it.

You can also select a set of folder and files and do a regular expression search on them. All results will be displayed in the code explorer window.



## ***Log Window***

In OptiPerl you can have a live view of the Logs created by the web server either internal or external.

These logs exist:

- Access Log. This shows the files server by the server. If you have an external server loaded, then it lists the file "Access Log File" in the Options Dialog.
- Errors Log. This shows the errors encountered by the server. If you have an external server loaded, then it lists the file "Error Log File" in the Options Dialog.

Right clicking on the window enables you to make it stay on top of the application.

## **Main Editor**

The main editor of OptiPerl is a sophisticated editor with syntax highlighting. It is tightly integrated with the rest of the environment, and highly customizable from the [Options Dialog](#).

You can do the following from here:

- Right clicking accesses a menu where you can control bookmarks, search a word in [perl documentation](#), open the [To-Do list](#).
- Pressing CTRL-J invokes the [Code Templates](#).
- Clicking on the gutter will add or remove a breakpoint. Read more in [Debugging](#).
- Pressing F1 on a key word will bring up [perl's core and module documentation](#).
- Holding down Control and moving the mouse cursor on a bracket character `{{(<>)}}` will show it's counterpart in the editor.



## ***Parameter List***

The parameter list dialog is used when running your script in the console or selecting parameters in the [external tools](#).

The edit box shows the parameter going to be used, while the list view some predefined tags and their value (at the time the dialog was invoked). The hint of the parameter edit box previews the parameters after processing the tags.

`%file%`

The filename of the script.

`%filenoext%`

This is the filename of your script excluding the extension.

`%folder%`

The folder in which the script is, excluding the trailing backslash.

`%path%`

This will be replaced with the full path to the script you are editing.

`%pathsn%`

Full path to script as a dos short name (no spaces).

`%querybox%`

The query in the query box.

`%query%`

Prompts user to enter some text.

`%selection%`

The text selected in the editor. If the text spans many lines then the EOL characters are deleted.

`%url%`

Sends the same url that is called in the internal browser when running the script.

`%word%`

The current word under the cursor.

## ***Perl Core and Module Documentation***

Included with OptiPerl is all of Perl's Core and Module documentation, organized with topics and keywords. You can also search for custom queries. Open the help manually from Help/Perl Documentation, or search for the keyword under the cursor in the main editor by pressing F1.

## ***Perl Information***

Shown here is the information about the build of Perl you are using. The "Detailed Information" button toggles between the information extracted with the `-v` and `-V` switch.

## ***Perl Printer***

Perl printer is a two-panel window to convert raw text to print statements.

Enter the text to be outputted in the left editor, or open a file, and the text is converted to legal print statements.

You can select whether the print statements will be a "here-document" format, or select a start and end command and quote for each line.

### ***Perl tidy interface***

Perl tidy is an interface to Steven L. Hancock's perltidy (included with OptiPerl's distribution). His excellent perl script can be used to reformat perl code to make it more readable. The bottom part of the window shows a preview of your script, after the process. By pressing save, you can save your changes.

It is highly recommended to have the "Backup" option enabled before pressing save.

## ***Pod Extractor***

Pod extractor enables you to view POD files, or embedded pod information in a script. When selecting Pod Extractor, the window is opened with the filename of the script you are editing to be opened. Press "Extract" or select first a different file by pressing the Open File button.

OptiPerl has the only internet browser with a find dialog that supports regular expressions. To invoke the dialog, press "Ctrl-F"

Another useful feature is creating a list of all files in your perl directory that contain pod information. Select "Files" and then enter a folder to search in. Recommended is \perl or \perl\lib. This will list all the files in the combo box permanently so you can select them easily. You can refine the search any time later by pressing "Files" again.

## ***Project Options***

Setting project options are necessary to correctly publish your project.

Select here:

- **Host:** An FTP Server to upload, for example `ftp.mypage.com`
- **Username:** Username to login.
- **Password:** Password to login. This is saved encrypted in the project's file.
- **Starting Path:** Where publishing will begin from, for example `cgi-bin\` or `cgi-bin\messageboard`
- **Change shebang:** When uploading a perl script that uses a she-bang line, how to change it while uploading.
- **Version convert to server:** If your script uses version converter tags, whether they will be moved to "server".
- **Internal server root-path:** The same as selecting one from the menu Server/Internal server root when using the internal server. This is used to quickly associate such a path with a project, so you don't need to select another on each session of OptiPerl.

See also "[Working with Projects](#)"

## Query Editor

The query editor is called when you press the Edit button in the main window.

This brings up two ways to edit the query:

- Using the query editor you can make up queries like "fname=George&email=george%40mail.com". The first row is the Name, and the second is the Value. Use this to emulate form components of a web page.
- You can edit the string manually to take any other form.

When debugging CGI scripts that have input from html pages, you cannot load the html page, press the submit button and enter the debugger. You can however debug using a query entered in the query box.

To help with this, press the "import" button and select the html page that has the form data. The html code will get parsed into the query editor, so you can emulate the query when using the debugger.

There are several methods you can select for sending the query:

- **GET:** Sends the query to the script using a get method `$ENV{ 'QUERY_STRING' }`. This is the same as running a url in the form: `http://127.0.0.1/cgi-bin/test.cgi?myquery`. You can use the GET method for running CGI scripts in the internal, external or secondary browser or debugging.
- **POST:** Feeds the query with a post method. You can use the POST method for running CGI scripts in the internal or external browser or debugging. The input can be read by the script from `<STDIN>` or by using the CGI module.
- **Path Info:** Sends the query as path info ( `$ENV{ 'PATH_INFO' }` ). If you want to include path info and also a GET query, then select Path Info as the method and enter a query like the following: `pathinfo?name=value`. You can use Path Info for running CGI scripts in the internal, external or secondary browser or debugging.
- **CMD Line:** Specific to console scripts. Sends the query as a command line ( `@ARGV` ) to console scripts, when run in the web browser text tab.



## **RegExp Tester**

With the Regular Expression Tester, you can enter regular expression and test quickly the way they match, substitute and transliterate text.

There is also an Explain box, that uses Jeff Pinyan's *YAPE::Regex::Explain* module to analyze the pattern of the regular expression. This module is included with OptiPerl's distribution.

Enter some text in the edit box and some input for processing in the left panel. Updating takes place automatically.

In the explain list box, moving through the tokens also highlights in the edit box the corresponding part of the regular expression that is explained.

Here are some examples that you can enter. After entering, type some corresponding text in the left panel to get the results.

```
/(.+)\@(.\+)\.(.+)/
```

**\*Very\*** simple check for a valid e-mail. Notice if you enter in the input only one line, then in the output panel you will also get the back references found.

```
tr/A-Z/a-z/
```

Convert all uppercase to lowercase

```
m/(\d{3})-(\d{3})-(\d{4})/
```

Check input if it is a valid U.S. telephone number.

```
s/%([a-fA-F0-9][a-fA-F0-9])/pack("C", hex($1))/eg
```

This will unpack a URL Encoded String.

**Important:** You must include a starting and ending slash in the pattern you type. Entering a raw pattern does not work.

Regular Expression Tester is a program in it self! It is designed to quickly test implementations of regular expressions without endless rounds of editing-fixing-running.

## ***Search files***

The search files dialog has options for regular expression searching in the File Explorer. You can enter a pattern to search for, a file mask of files to be searched, and also whether the search should be case sensitive.

A special feature is "Search in binary files". When this is clicked then files that are binary will be search also for the regular expression you entered. For example, try searching in all \*.exe files in your windows folder for e-mails. The results whill be shown in the code explorer that can handle display of binary files.

## ***Secondary edit window***

The secondary edit windows are windows that can be opened and linked to files open in the main editor. The editors will link to the same file and not copies of the same file. This means that changing text in one will automatically change the text in all windows that are linked to the same file.

If you have the option "Synchronized scrolling" on then also scrolling is linked. This can be useful when comparing two files.

OptiPerl however has a [File Compare](#) tool.

## FTP Sender

Using the script uploader, you can quickly upload your script to your web page, and change it's CHMOD. You can also automatically change the shebang line depending on the server you are uploading.

The shebang is the first line in your script that looks something like this:

```
#!/usr/local/bin/perl
```

This line shows the location of perl on the internet server. Since the location of perl is different between servers, you have an option of selecting which shebang line will be used for each server. This way whenever you upload your script, the first line is changed to what you have selected (and afterwards restored). Or you can just leave empty so the script is uploaded as it is.

Note that offline, the shebang line is not important when testing scripts with the internal server. However if you are using an external server, you might need to change it on the fly when uploading and then restoring it back.

You can add to the database many servers and directories, to have handy if you upload your scripts to various servers and directories.

To add a server, do the following:

1. Press the "+" button to add a new record
2. Enter a host, for example ftp.mypage.com
3. Enter the port, the default is 21
4. Enter the directory for the script to be uploaded, like "/cgi-bin/"
5. Enter the CHMOD for the file, usually the default given is ok.
6. Enter the username for login to the script, like "myname"
7. Optionally enter the password.
8. Change the shebang line used at the server, or empty the field. The line you type here replaces the shebang line in your script (after the upload the original is replaced). If you leave this field empty, then no change occurs in the first line. Note that you must already have a shebang for the replacement to occur. If not (for example in modules) then the first line is not changed.
9. Add some notes if needed about the server.
10. Press "v" to save changes to the record.

The database is saved with encryption to your hard disk. However if you want increased security, don't enter the password field. This way you will be prompted only for the password whenever you upload.

By pressing "Send" the script is uploaded to the selected server, and the progress is reported.

Under the database, there is a standard database navigator set of buttons:



- "+" Add's a new record
- "-" Deletes a record

- "Up arrow" edits the selected record
- "v" Saves changes from a previous edit
- "x" Cancels changes from a previous edit

Note that some servers automatically change the chmod for scripts that are uploaded, so you might get a message "Could not CHMOD, but script uploaded". This is ok.

**Important: If a script already exists with the same filename, it is overwritten without warning.**

Some web hosting services do not offer CGI Script processing. You may want to contact your service to ask at which directory you can send your custom scripts.

## ***To-Do List***

The to-do list is useful to create statements of future enhances to your script. They are saved with the script, in a \*.op file.

To add a new to-do item, do the following:

1. Right-click on the window and select "add item"
2. Enter the action title, priority, owner, category and notes.
3. Click "OK"

This way you can add new items. Clicking the checkbox, also makes pending items done.

Clicking the title buttons, sorts the list according to the corresponding field.

## ***URL Encoder***

With this dialog you can Encode and Decode strings with the following methods:

**URL:** Encode string characters using the hex escape syntax of RFC1738. Non-alphanumeric characters other than period, dash and underline are replaced by '%' followed by a 2 digit ASCII hex code.

**Base64:** base64 encoding compatible with Internet protocol RFC 1521 (the MIME standard). Data encoded in this manner is safe for Internet, database and mainframe storage and use. By definition, the resultant string is at least 1/3 longer than the original

**UUEncoded:** Encode binary Source string using Unix-to-Unix encoding. Encoding increases the overall length of string by 1/3. This encoding technique has never been formalized and as a result, a number of different variations exist.

**Crypt:** Encrypts a string exactly like the crypt function in perl. The salt used is random (you can only encode, decode will not function).

## ***Watches Window***

In the watches window, you can add perl expressions that are evaluated while debugging. Right click on the window to add/edit/delete a watch.

While debugging, changing a value in the "value" column, will also change it in the program's flow.

Read more about this important debugging feature in "[Debugging](#)"



## **Web Browser**

The Web Browser in OptiPerl is the window that you can view all the output of perl scripts and html documents.

In conjunction with the Browser toolbar, it is a basic web browser. However it has an important feature just for debugging:

At any time, all html output can be viewed either as a web document, either as text. Also there is a "send" line, for sending input to <STDIN> when debugging scripts.

Read more in "[Debugging Scripts](#)" and "[Running Scripts](#)"

When running a script in the browser, there is a major difference on how the script is run, depending if "Run with Server" is selected:

- If "Run with Server" is disabled, then the script is run directly and all it's output is piped to the browser.
- If "Run with Server" is enabled, then the script is launched by sending a target URL to the browser, for example

```
http://127.0.0.1/cgi-bin/test.cgi
```

Also from the Web Browser window, under the "Auto-View" tab, you can view at real-time the changes that happen to a file you select.

OptiPerl has the only internet browser with a find dialog that supports regular expressions.

## ***Options Window***

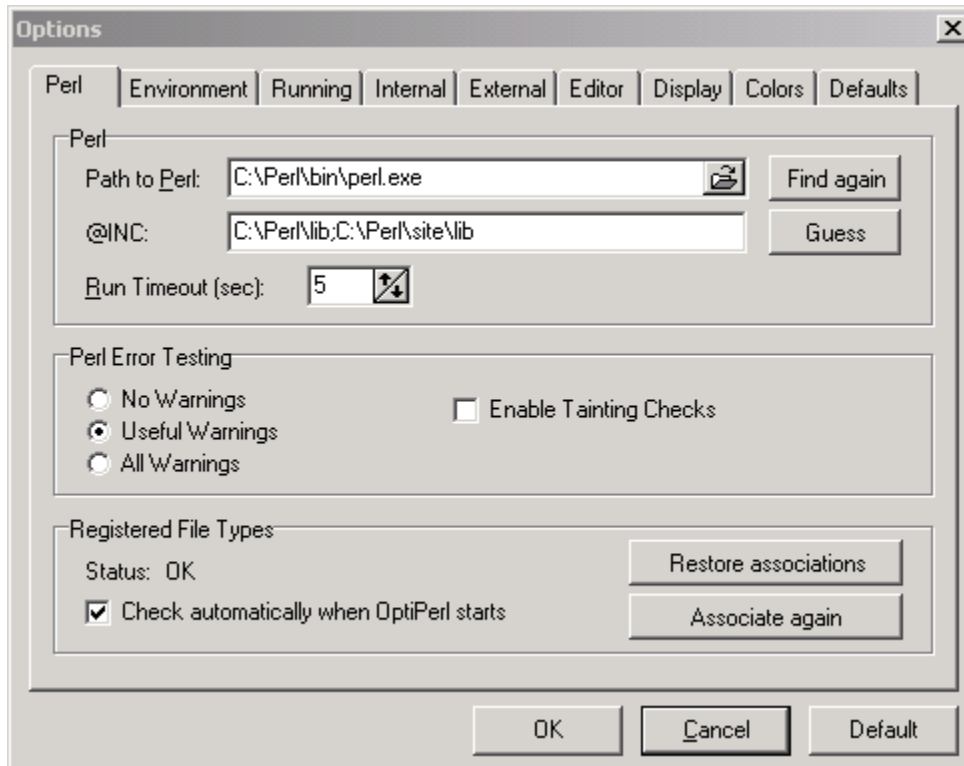
The options dialog has a large amount of options the cover many aspects of OptiPerl.

If you make some changes that may cause running or debugging to not work, you can press the "default" button to get correct options that depend on the path to perl found.

All controls have hint windows that explain each function.

## Perl

The perl tab affects the perl interpreter and how perl files are associated with OptiPerl.



The path to perl is important to be correct, for most of OptiPerl's features to work. Change the above if you have installed Perl in a different directory.

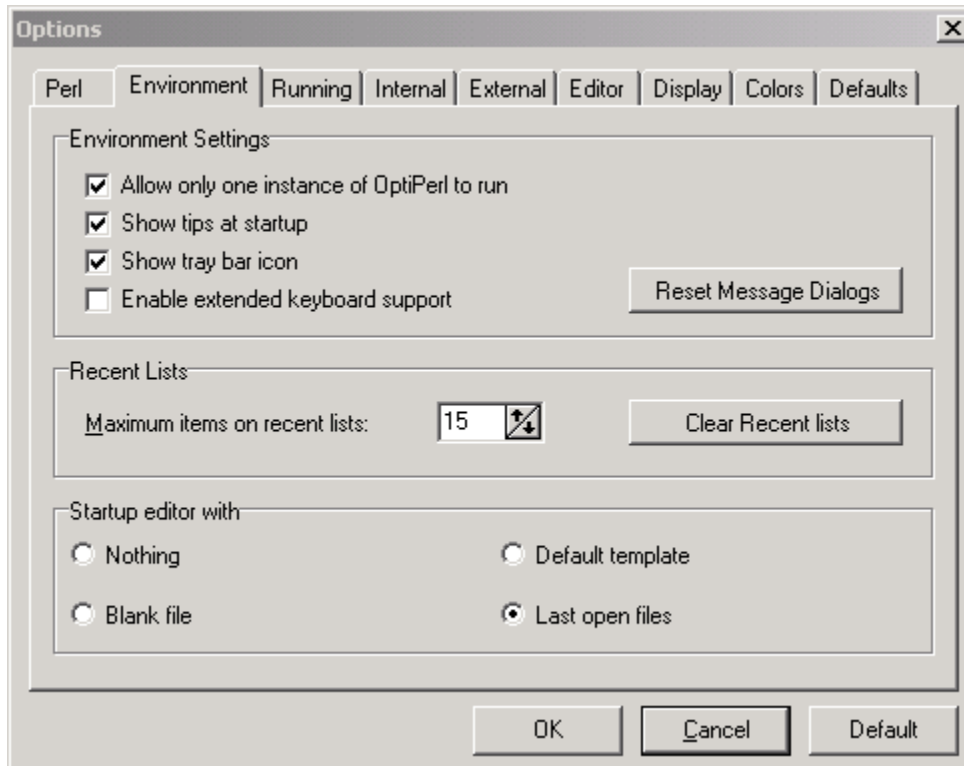
Timeout is the number of seconds OptiPerl will wait if you run a perl script, until it asks for confirmation to terminate it.

You can also select here the warning level reported when you test for compilation errors.

Registered file types with OptiPerl are the extensions pl, plx, cgi, pm. You can have OptiPerl check each time it starts if these are associated with the program, or restore them to a state as before OptiPerl was installed.

## Environment

Selections that affect OptiPerl' s environmet.



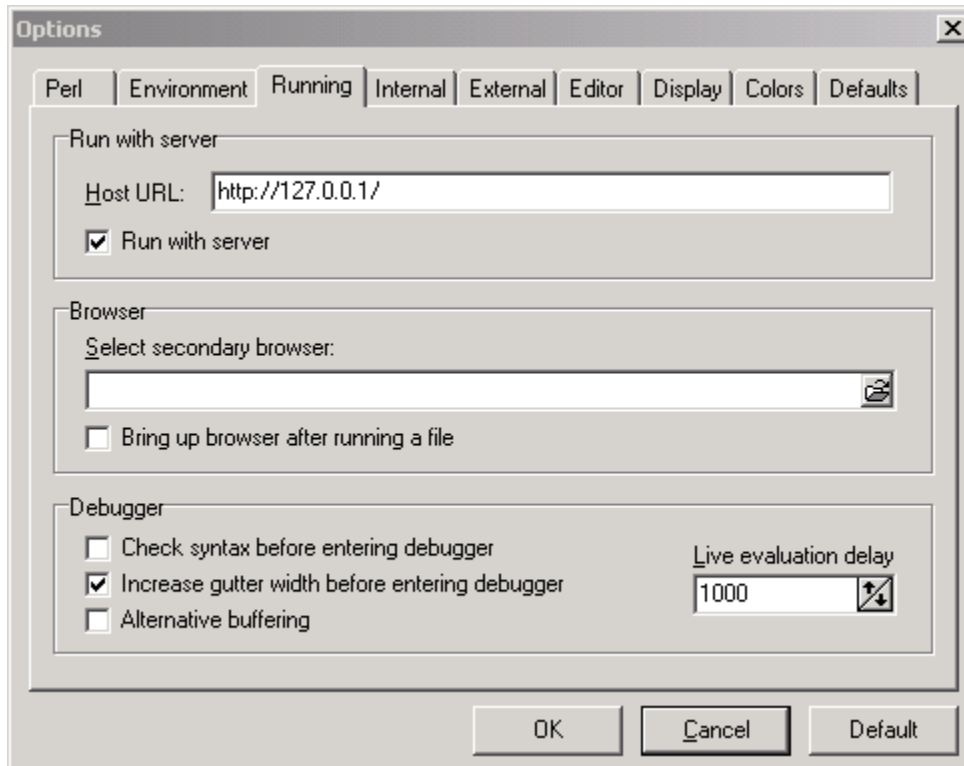
By pressing "reset message dialogs", all dialogs that had a "do not display again" check box will show again.

Recent lists number and resetting affect all recent lists used by OptiPerl, in the File and Project menu, and also in the drop-down list's of edit boxes used in the program.

You can also select how the editor should start up when OptiPerl is loaded.

## Running

Run tab affects running and also debugging.



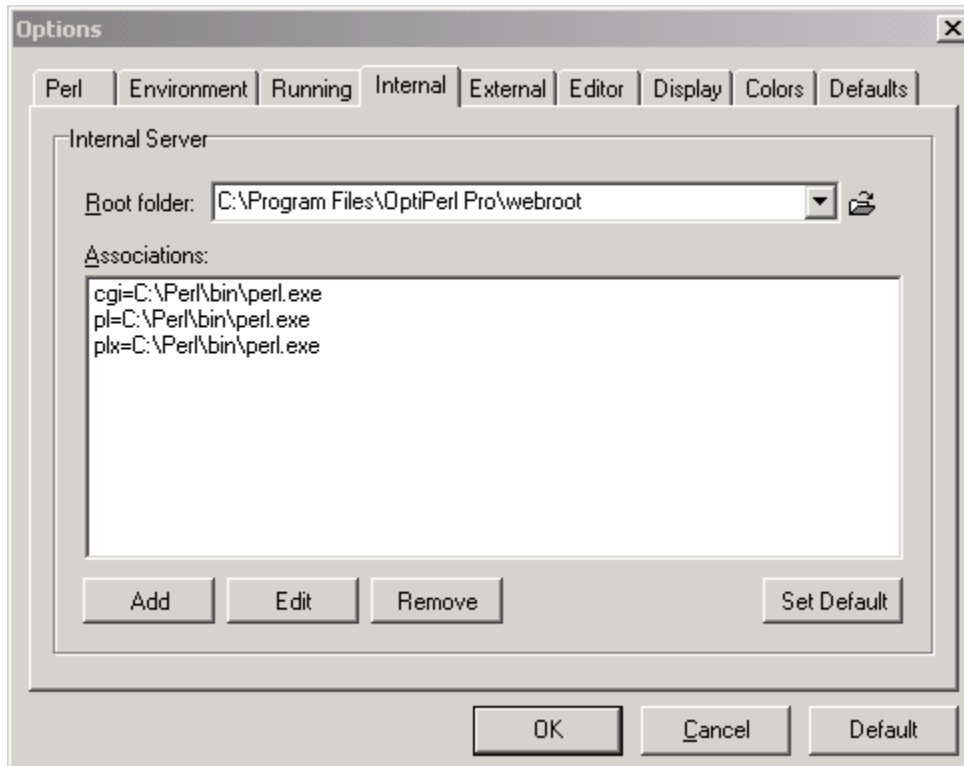
The host URL is the URL added in front of URL's sent to the browser when running scripts, this probably will not need any change. Run with server is whether a http URL will be sent to the browser, or the file URL of the scripts output.

Read more in

- [Running Scripts](#)
- [Internal Server](#)
- [Running without a Server](#)
- [Using an External Server](#)

## Internal Server

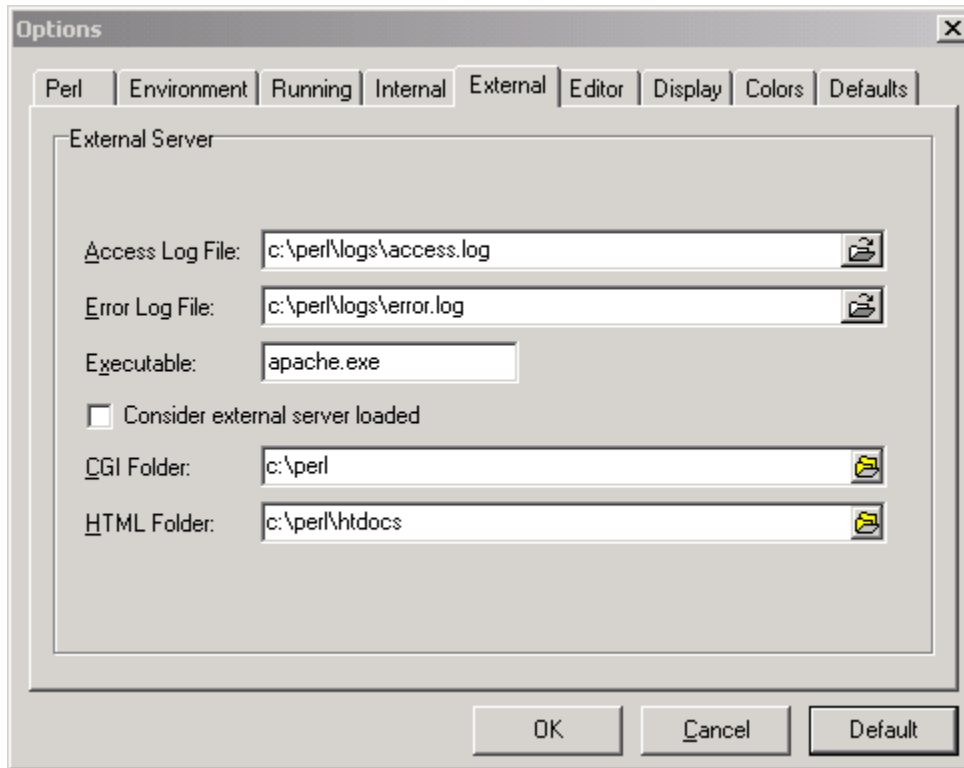
Internal server settings and associations.



This page has settings that have to do with the internal server when enabled. The root folder is the path that the internal gets files from when you give it a http request. For example, <http://127.0.0.1/index.html> will return the index.html file if located in this folder.

## External Server

External server settings and folders.

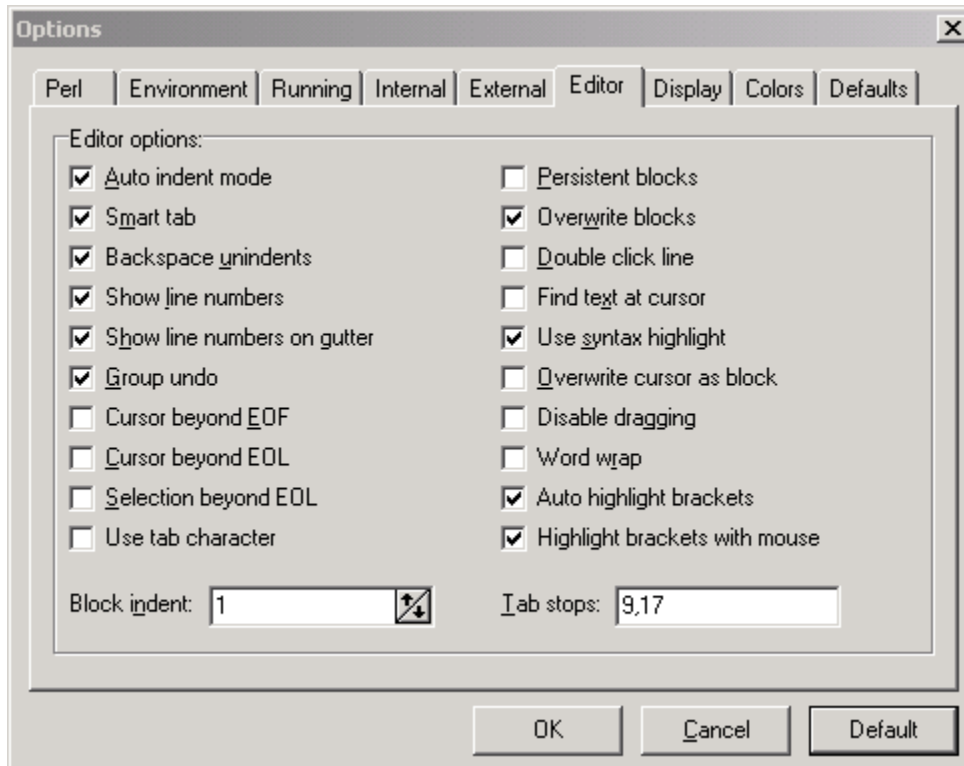


This page has settings that have to do with an external server. Read more in:

- [Running Scripts](#)
- [Using an External Server](#)

## Editor

Editor options.

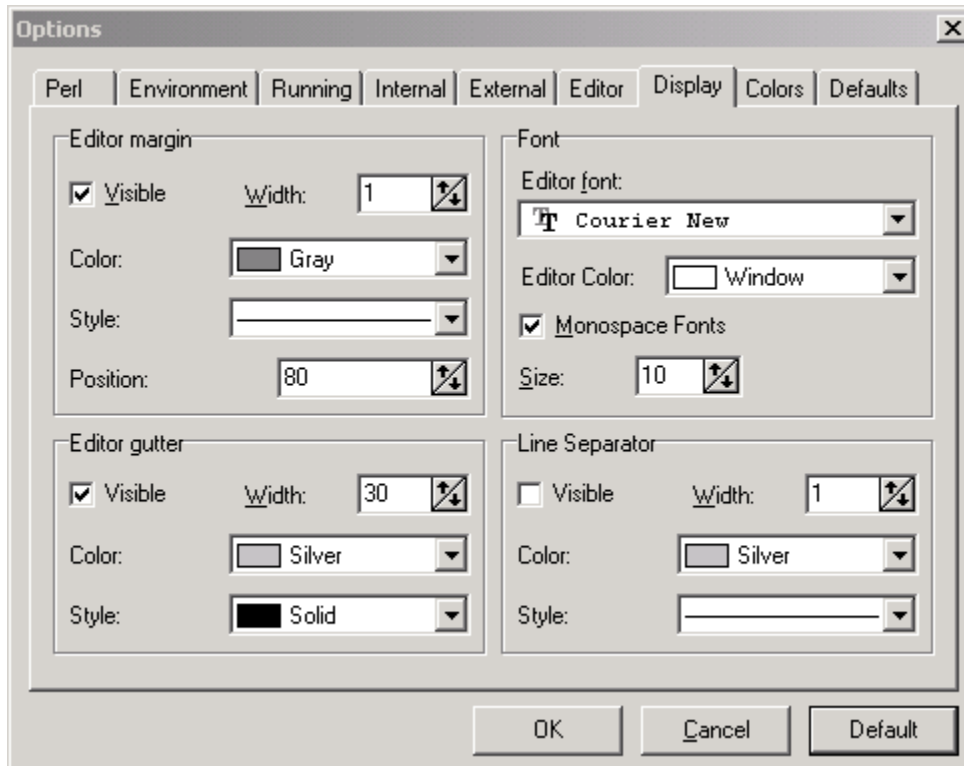


The editor tab contains options that affect the behavior of the editor. Hover the mouse over each of them for a detailed description.



## Display

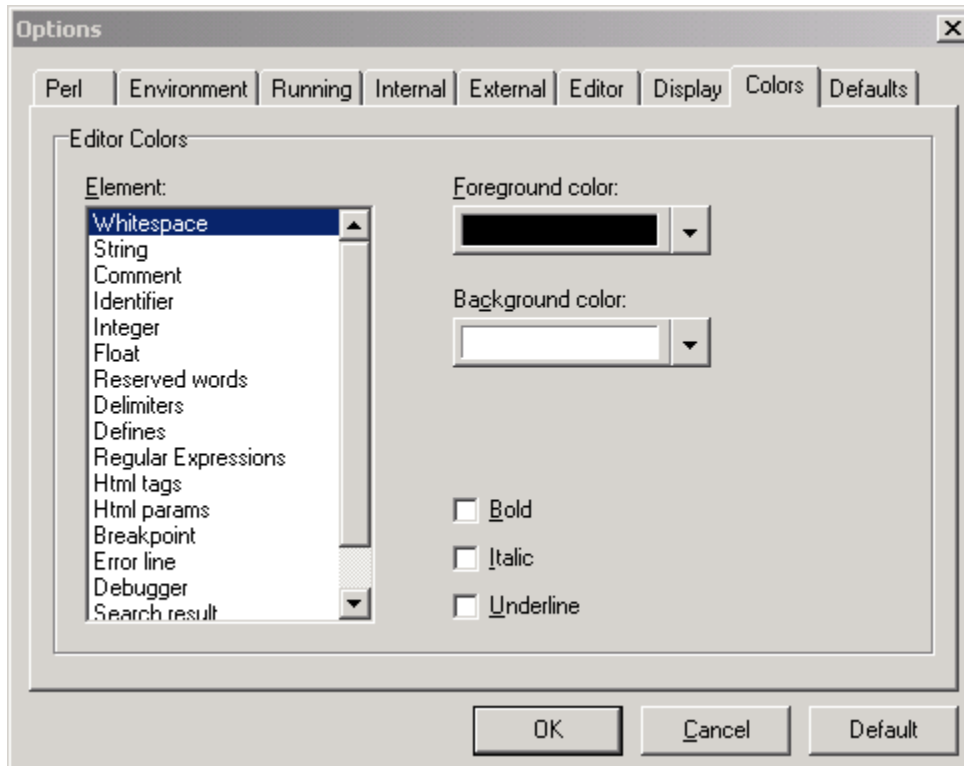
Display elements of OptiPerl' s editor.



The display tab affects the display of the editor. You also select a font for the editor (a monospace font is recommended).

## Colors

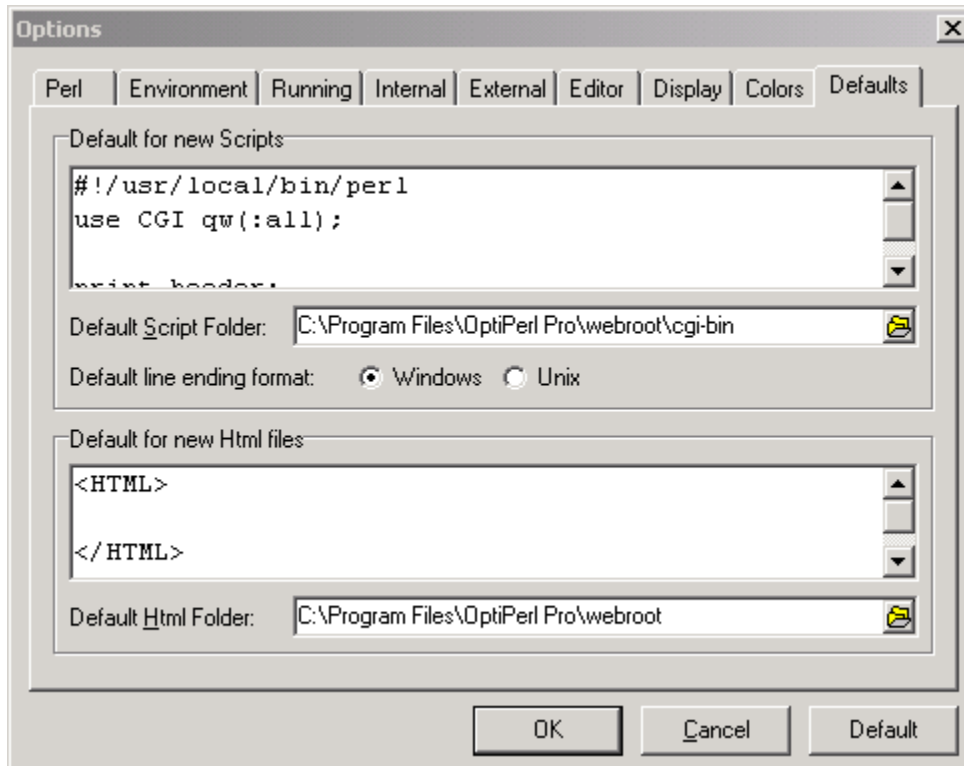
OptiPerl' s colors.



Here you can customize all the colors of the main editor, including the color of syntax highlighting and status lines.

## Defaults

Default text when creating new files



Here you can enter default scripts when you create new files, and also the default position of them. Note that in this folder the new files are temporarily saved also if you try to run them without saving them first.

The default line ending affects the default way scripts are saved.

