

NetCloak

User's Guide

*“Dynamic, Interactive Web Sites...
... No Programming Required”*

rev. C

Maxum Development Corporation
Post Office Box 315
Crystal Lake, IL 60039-0315

Disclaimer of Warranty and Limited Warranty on Media

Maxum will replace defective distribution media or manuals at no charge, provided you return the item to be replaced with proof of purchase to Maxum within 60 days of product purchase. The limited warranty shall continue for the replacement software the remainder of the 60 days or for 30 days from the date you receive the replacement, whichever is longer. Maxum does not warrant that the software is free of bugs, errors, and omissions.

Maxum makes no warranties, expressed or implied, including without limitation any implied warranties on the media and manuals, including warranties of merchantability and fitness for a particular purpose. Maxum makes no warranty or representation, either express or implied, with respect to NetCloak, its quality, performance, merchantability, or fitness for a particular purpose. As a result, NetCloak is licensed “as is” and you the purchaser are assuming the entire risk as to its quality and performance.

In no event will Maxum be liable for direct, indirect, special, incidental, or consequential damages resulting from any defect in NetCloak or its documentation, even if advised of the possibility of such damages. In particular, Maxum shall have no liability for any programs or data stored using NetCloak on data storage devices of any description, including the cost of recovering such programs or data.

THE WARRANTIES AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS ORAL OR WRITTEN, EXPRESSED OR IMPLIED. No Maxum development distributor, dealer, agent, or employee is authorized to make any modifications, extensions or additions to this warranty. Some states do not allow exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. In any case whatsoever, regardless of the form of action, Maxum liability to you for actual damages will be limited to the purchase price you paid for the software.

Copyright Notice

The NetCloak documentation, program design, and design concepts are copyrighted, with all rights being subject to the limitations and restrictions imposed by the copyright laws of the United States of America. Under the copyright laws, this manual may not be copied, in whole or in part, including translation to another language or format, without the express written consent of Maxum Development Corporation.

ABOUT THIS GUIDE.....	1
If You Need Help.....	1
WHAT IS NETCLOAK	2
SYSTEM REQUIREMENTS.....	4
QUICK START.....	5
Installing NetCloak.....	5
Creating Dynamic “Cloaked” Pages.....	6
Conditional Pages.....	10
More Useful Commands.....	17
ADVANCED TECHNIQUES.....	20
Cumulative Effects of Commands	20
Using Negative Comparisons.....	22
Combining Commands by Negating.....	23
Using Nested INSERT Commands.....	24
MACROS.....	26
Defining Macros.....	26
Using Macros	27
Changing Macros.....	28
SERVING CLOAKED FILES	29
Cloaked File Limitations	29
NetCloak Caching and Aliases	29
USING NETCLOAK WITH OTHER SERVER TOOLS	33
Calling other Plug-Ins via PIXO	33
How Back-Ending Works.....	34
Defining Back-End Processes	34
Troubleshooting Back-End Setups.....	36
PROCESSING HTML FORMS WITH NETCLOAK.....	37
Before You Get Started.....	37
FORMS PROCESSING TUTORIALS.....	38
Forms Processing Basics.....	38
Forms to E-Mail.....	42
Guestbook Tutorial: Adding to Existing Pages.....	46
Recipes Tutorial: Creating New Pages.....	50
Creating Pages: The Feedback Example	60
USING AUTOLINK.....	65
USING RESPONSE FILES.....	66
USING RECENT ARTICLE LISTS.....	67
Using Recent Articles Lists.....	67
Maintaining Multiple Recent Articles Lists.....	68
MAIL ACTIONS	70
What Are Mail Actions?.....	70
Setting Up Mail Actions.....	70

CONTENTS

- Creating a Mail Action FDML File 74
- Debugging Mail Actions..... 75
- Message Content Types..... 76
- Update Recent Database..... 77
- INSTALLATION 78
 - An Overview Of Suffix Mappings And Actions..... 78
 - Installing NetCloak..... 79
 - Uninstalling NetCloak..... 82
 - Installing NetCloak with WebSTAR 83
 - Installing NetCloak with AppleShare IP 83
 - Installing NetCloak with WebTen..... 84
 - Installing NetCloak with Other Web Servers 87
 - Installing NetCloak Pro with Personal Web Sharing..... 88
- UPGRADING NETCLOAK..... 89
- CONFIGURING NETCLOAK..... 91
 - Files..... 92
 - Options..... 96
 - HTML..... 100
 - Logs..... 103
 - Security..... 104
 - Configuration Outside Of NetCloak..... 107
- EDITING COUNTERS, GROUPS AND GLOBALS 111
 - Resetting Counters Automatically 112
 - Simplifying Maintenance With Groups..... 114
 - Accessing Global Variables 117
 - Using AppleScript to Access Global Variables..... 119
- USING THE NETCLOAK CGI..... 122
 - Using NetCloak Without Suffix Mappings..... 122
 - Installing the NetCloak CGI 123
 - Configuration 126
 - Global Variable AppleEvents..... 127
- INDEX..... 128

This manual is the NetCloak User's Guide, and contains an introduction to NetCloak, instructions for installation and configuration, and tutorials on using NetCloak. It is intended to guide new users through the installation process and acquaint them with using NetCloak's dynamic page and forms-processing commands. The User's Guide tutorials are intended to teach methods and concepts, and so do not cover every single command. The companion Reference Guide contains a comprehensive listing of commands and their use and syntax.

Each guide covers dynamic-page and forms processing commands, even though the forms processing commands are only supported in NetCloak Pro. This is mentioned in appropriate sections, but the user is assumed to be familiar with this distinction.

This manual does not explain how to install or configure a web server or how to create HTML pages. For more information on setting up a web server or writing HTML, see the server software documentation or an HTML reference.

If You Need Help

There are three ways for you to get help if you have a problem.

1. The place to start is Maxum's Web site:

<http://www.maxum.com/>

Here you will find: software updates, complete on-line documentation, Frequently Asked Questions (FAQs) and other timely information.

2. If the answer to your question is not at our Web site, you can send e-mail to our Technical Support staff:

support@maxum.com

We'll get your question answered quickly. (in addition to replying directly to you by e-mail).

3. Finally, you might want to subscribe to the NetCloak mailing list. For information, go to:

<http://www.maxum.com/maillists.html>

The quality of the comments and help on this list are exceptional.

WHAT IS NETCLOAK

NetCloak is an add-on tool for Mac OS-based web servers that lets you create dynamic HTML pages. NetCloak runs in conjunction with your web server software and allows you to improve security, simplify site development and maintenance, and control exactly how a page looks to an individual client.

With NetCloak, you can create HTML documents that change every time they are viewed by a client. You will be able to show and hide portions of your HTML documents using a variety of different criteria, as well as add dynamic information to your pages. NetCloak is simply the best way to set your Web site apart from all the others on the Web.

Here are just a few of the hundreds of possible things you can do with NetCloak:

- Display random graphic banners or text
- Hide links from people outside your intranet
- Easily add counters to any or all of your pages
- Control search engine Web crawlers
- Dynamically insert information into a Web page
- Customize content or formatting for Mac or Windows
- Automatically redirect users accessing obsolete pages
- Automatically insert the date each page was last updated
- Maintain headers and footers for your entire site in one file

NetCloak also lets you insert a variety of information into your documents, including the user's TCP/IP address, browser, username and more. You can also use NetCloak to create macros for text used repeatedly on your Web pages, making them easy to create and maintain.

NetCloak does all this with minimal effect on the performance of your server by giving you:

- Configurable document caching.
- Configurable security and logging options.
- Selective cloaking of only the documents you choose.
- Plug-In operation on WebSTAR API-compliant servers.

NetCloak Pro adds powerful forms-processing and e-mail features for professional webmasters. NetCloak Pro can take data from HTML forms and use it to automatically:

- Create new HTML pages.
- Insert text or HTML into existing pages.
- Store data in tab-delimited text files for later processing by virtually any application (including databases, spreadsheets and text-processors).
- Send e-mail messages.

In addition, using Mail Actions, NetCloak Pro can perform all of these same tasks in response to mail messages received by any e-mail address.

Here are just a few examples of what NetCloak Pro can do:

- Create interactive, threaded discussion forums (bulletin boards, job postings, classified ads, announcements, etc.).
- Allow users who know nothing about creating Web pages to add content to a Web site, including their own home pages, news articles, project or classroom updates, and more.
- Automatically maintain a Guest Book where users can sign in as well as read about people who have visited your Web site.
- Accept product orders and other requests.
- Enable users to automatically subscribe to e-mail lists or send feedback via e-mail messages by filling out an HTML form.
- Create a Web archive of the messages posted to an e-mail list, or merge an e-mail list with a Web-based discussion forum.

NetCloak Pro, and your creativity, will enable you to generate flexible, easy-to-use forms to make your web server stand out from the crowd.

SYSTEM REQUIREMENTS

NetCloak runs on Mac OS computers, including the Apple Macintosh and Macintosh clones. NetCloak has the following additional requirements:

- System 7.5 or higher. (System 7.1 with Thread Manager)
- Open Transport or MacTCP.
- 2 MB free RAM.
- 2 MB free hard drive space.

Mac OS web server software is also required (see your web server manual for additional server requirements).

To access your server when it is up and running you will need a Web browser such as Netscape Navigator or Internet Explorer.

For network requirements consult your web server manual.

Your NetCloak software disk includes the NetCloak Installer which will place the NetCloak 3.1 folder on your hard disk.

The NetCloak 3.1 folder downloaded from the Internet or created by the NetCloak Installer contains:

- READ ME document with instructions on getting started.
- Upgrading document with information about upgrading from previous versions of NetCloak.
- NetCloak Plug-In
- NetCloak.acgi, the CGI application.
- NetCloak Introduction folder with introductory tutorials and samples, including the User's Guide in HTML format.
- Sample Scripts folder with AppleScripts that demonstrate how to access NetCloak global variables.
- Goodies folder with utilities and AppleScripts to make using NetCloak with other third-party software even easier.

Installing NetCloak

To quickly get acquainted with NetCloak, you can install the NetCloak Introduction as described below. The Introduction includes links to sample pages which demonstrate some of the features of NetCloak and the on-line User's Guide (in HTML format). The tutorials are included here so you can read them either online or in hard copy.

The instructions below assume that your web server supports WebStar API plug-ins and is already configured for Web service. If it is not, consult your web server documentation for instructions. If you have trouble with these instructions, see the Installation section of this guide for more detailed instructions for installing NetCloak with various web server applications.

To Install The NetCloak Introduction:

1. Expand the compressed NetCloak demo package, or insert the NetCloak disk and run the installer to install the NetCloak 3.1 folder on your hard disk.
2. Copy the NetCloak Plug-In to your server's plug-ins folder.
3. Open the NetCloak Introduction folder, and copy the entire contents to your web server's root folder. If your web server software supports multiple root folders, copy it to your "default" or "main" root folder.
4. Start up the web server, open your browser and load the following URL:
`http://your.server/NetCloak.nc1k`

You'll need to substitute the address of your Web server for "your.server", of course.

The NetCloak Demo and Information page should appear, with some of the basics about NetCloak, examples of what you can do with it and tutorials to get you started using NetCloak yourself. The tutorials are also included here in the User's Guide. Flip forward to the tutorial section and you can follow along in the book.

The Installation section in this guide has detailed and specific instructions for installing NetCloak with various Macintosh web

QUICK START

servers. If the simple instructions above do not work, refer to the detailed steps described in the Installation section.

Creating Dynamic “Cloaked” Pages

With NetCloak installed, you are now ready to add cloaking commands to pages on your Web site, as described in this section.

We strongly recommend that you be completely comfortable writing HTML pages before you start using NetCloak. If you are not familiar with HTML you can get more information from a variety of sources. In addition to numerous books on HTML, there are many tutorials available on the Web. Once you are familiar with HTML, adding NetCloak features to your pages is easy.

Using NetCloak Commands in Your Pages

The NetCloak plug-in will automatically configure your server so that any page ending in “.ncl” will be processed by NetCloak. In other words, once you have dropped the NetCloak plug-in into your Web server plug-ins folder and restarted the server, all “.ncl” files will be “cloaked”. (In addition, all forms with an action URL ending in “.FDML” will automatically be processed by NetCloak Pro.) You can also choose to have NetCloak process all HTML pages on your server by setting the “.HTML” suffix mapping within your Web server to be processed using the “CLOAK_PI” action. See the “Installation” section of this guide for complete details.

Inserting Dynamic Information

NetCloak provides new commands that you can incorporate into your HTML pages to add dynamic, interactive elements to your site with remarkable ease. While NetCloak commands are not standard HTML tags, they are processed entirely on your web server, so they work with any Web browser.

All NetCloak commands begin with an opening angle bracket (“<”), just like HTML tags. Commands are closed with a closing angle bracket (“>”). The NetCloak command name always immediately follows the opening bracket. Command names are not case sensitive, but are generally written in all uppercase by convention; this makes them easier to recognize in HTML pages.

The best way to start using NetCloak is by working with NetCloak's INSERT commands, which are used to add dynamic information into any cloaked page. For example, to add the current time and date, simply place the commands `<INSERT_TIME>` and `<INSERT_DATE>` in the text of any cloaked page, as in:

It is `<INSERT_TIME>` on `<INSERT_DATE>`.

NetCloak INSERT commands all display, or insert information from the server into the page in place of the command. For example, when loaded by a Web browser, the first sample line shown above will appear something like this:

It is 12:47 on December 14, 2000.

The time and date displayed will change every time the page is loaded, and will always reflect the current time and date on the server.

Similar commands allow you to dynamically display the user's name or IP address, cookies or user-entered variables and other information about the client, page, request or server.

Many NetCloak commands have required or optional additional parameters. In other words, additional settings that specify differences in how the information displayed by the INSERT command should be shown. Parameters must be specified in order, according to the command description in the Reference Guide, and are separated by one or more spaces.

For example, the `<INSERT_DATE>` command offers two optional parameters, one for a "time offset" and one for the desired format. The offset allows you to specify a number of hours difference between the actual date and what is displayed. The format parameter allows you to override the default display format and display the date as either a short date or long date. For example, to show tomorrow's date as a short format, you would write:

In 24 hours, it will be `<INSERT_DATE +24 SHORT>`.

On a web page, users will then see something like:

In 24 hours, it will be 12/15/00.

QUICK START

What is noteworthy is that parameters can be added after the command name, are separated by spaces, and are inside the angle brackets around the command. Parameters for each command are something you will learn later as you start using more NetCloak commands. See the Reference Guide for details on the parameters for each command.

More INSERT Commands

There are many different INSERT commands, dealing with date and time, counters, file information, and more. We will cover a few here to get you acquainted with the range of commands that are available.

Browser Information

Several commands allow you to display a variety of information about the browser. `<INSERT_CLIENT>`, for example, will display the name of the client Web browser, as reported by the browser itself. The command `<INSERT_DOMAIN>` displays the user's IP Address or domain name. Later we will see how to change the content of pages dynamically based on the same information.

Countdown

NetCloak also has a unique command which will insert the amount of time remaining until or elapsed since a specified date and time. The best (and all too common) example of this was the "Millennium Counter" displayed on many Web sites in 1999 that showed the time remaining until midnight December 31, 1999. Even though New Years Day 2000 has passed, the `<INSERT_COUNTDOWN>` command is still plenty useful. For example:

The big sale ends in just 6 hours!

or

Our plant has worked 158 days without a lost time accident!

You can use this command to count down the time until a holiday, meeting or other future event as well as counting time elapsed since a past one. The `<INSERT_COUNTDOWN>` command has options to

display the time period in minutes, hours, days, weeks or any combination of those units.

Counters

One of NetCloak's simplest and most useful insert commands is `<INSERT_COUNT>`. This command, as you would expect, inserts a counter into the page that is incremented every time the page is served.

The counter will be inserted as text in your HTML document. Any surrounding formatting tags will affect the display of the counter inserted by NetCloak.

You can create any number of counters simply by giving each one a unique name. By naming your counters you can have a separate counter for different pages, or refer to the same counter on more than one page.

A complete list of INSERT and other commands is in the NetCloak Reference Guide.

Create A Cloaked Page

You are now ready to create a cloaked page. First, we'll create a simple page from scratch, to show just how straightforward "cloaking" is.

1. Create a new document with a text editor, preferably BBEdit or SimpleText.

2. Enter exactly one line:

Accessed `<INSERT_COUNT>` times.

3. Save the file in your Web server root folder as "test.nclk".

4. Go back to your Web browser, open a new window, and access the URL:

`http://www.yourserver.com/test.nclk`

Of course, you must replace "www.yourserver.com" with the name or IP address of your Web server.

The browser should display the line as "Accessed 1 times." If you reload the page, the counter should increment automatically.

That's it! You've created your first cloaked page.

QUICK START

You can now start using NetCloak INSERT commands in pages on your Web site. Remember that NetCloak commands will only work in pages processed by NetCloak. All files with names ending in ".nclk" are automatically cloaked, so without additional server setup you can only use these commands in pages with the ".nclk" suffix. While you are evaluating the NetCloak demonstration version, we recommend that you name or re-name pages with the suffix ".nclk" when you use NetCloak commands in them.

NetCloak Tip: If you ever have problems with commands not working as expected, try using TagBuilder to add the commands to your page. By using TagBuilder, the text of the command will be dropped into your page for you, avoiding possible syntax mistakes and typos. It also provides a handy list of available commands which is ideal when you are learning to use NetCloak.

Extensions for creating NetCloak commands in your pages with BBEEdit or Dreamweaver are also provided in the NetCloak package.

Conditional Pages

While NetCloak INSERT commands are easy to use and a quick way to add new, dynamic features to your Web site, NetCloak's real power is in its ability to create conditional HTML. In other words, with NetCloak you can create pages that change in different circumstances- based on the user, browser or other conditions.

HIDE and SHOW Overview

Think of your Web page as a stream of water flowing through a faucet. This analogy isn't far off the mark, as HTML pages are actually nothing more than text "streams" sent by your Web server, which is the "faucet". NetCloak "SHOW" and "HIDE" commands allow you to control the stream within the page, stopping the flow of text and then resuming it later.

Here is an example of the simplest possible HIDE and SHOW logic:

```
This line is visible to web browsers.  
<HIDE>  
This line is not visible, even when users view the HTML  
source.  
<SHOW>
```

The first line in this example will be sent to any remote Web browser under any circumstance (assuming there were no prior HIDE commands in the page). The HIDE command turns off the stream of text going to the browser. After the next line, which will be hidden from all browsers, a SHOW command is used to turn the stream of text back on.

The HIDE and SHOW commands, as used here, are unconditional. In other words, the <HIDE> command will always turn off the text stream and the <SHOW> command will always turn it back on. By themselves, unconditional SHOW and HIDE commands are not very useful. Conditional SHOW and HIDE commands are much better, and we will see them next.

NetCloak Note: The analogy of a stream of water flowing through a faucet has a flaw when taken literally. When you turn a faucet off, the water stops flowing through the pipe, and when you turn it back on, the flow of water resumes from where it left off. This is not the case with NetCloak. When you use a NetCloak HIDE tag, the text stream to the browser is interrupted but the page continues to be processed at the server, and when a SHOW command is encountered, the text stream to the browser resumes from that point. Text between the HIDE and SHOW commands is lost as far as the browser is concerned.

Adding Conditions

NetCloak HIDE and SHOW commands that depend on various conditions are much more useful than the unconditional ones. The conditions that the commands will depend on are actually information on the server, like the time of day, the value of a variable, or client's IP address. In other words, NetCloak will compare the current value to some specified value (or values) and hide or show text following the command based on the result of the comparison.

Some simple uses for this include showing a different message in the page depending on whether it is morning, afternoon or night; including text (or hiding text) from a certain number of visitors or on a certain day; or hiding links on the page based on where the user is loading the page from. This section has a few simple examples to introduce the concepts, and then will let you discover more commands and uses for them by exploring other examples and the

QUICK START

Reference Guide. Once you are comfortable with the commands, you should also take a look at the more complex examples linked from the Introduction page.

Example: HIDE and SHOW by Time

You can use NetCloak's SHOW_TIME command to have your server recognize the time and greet a user according to the hour:

```
<HIDE><SHOW_TIME EQUALS 05 06 07 08 09 10 11>
Good Morning!
<HIDE><SHOW_TIME EQUALS 12 13 14 15 16>
Good Afternoon!
<HIDE><SHOW_TIME EQUALS 17 18 19>
Good Evening!
<HIDE><SHOW_TIME EQUALS 20 21 22 23 00 01 02 03 04>
Good Night!
<SHOW>
Here in Chicago, it is <INSERT_TIME SHORT>.
```

The HIDE and SHOW TIME commands are based on a 24 hour clock, and here we specify multiple comparison constants. By using several constants, the command will be triggered if the current hour is equal to any of the hours specified. In this case, the first line turns on the stream of text on when the current hour is 05, 06, 07, 08, 09, 10, or 11. In other words, the stream is turned on if the current time on the server is between 5:00 a.m. and 11:59 a.m., and the server will then say "Good Morning!".

In any case, the next to last line unconditionally turns the stream back on for all cases with a SHOW command so that the server will send the current time following the greeting.

You can use a similar technique with the HIDE_DATE command to show a message on specific date(s). See the Reference Guide for details on the HIDE_DATE command.

NetCloak Tip: Any time you are using conditional commands that don't seem to be working, add the corresponding INSERT command to the page at the point where the HIDE and SHOW logic appears. In the example above, if the greeting message is incorrect during testing, the INSERT_TIME command will show the current time, which will help to resolve the difficulty. For example, if the server's clock is set

incorrectly, the `INSERT_TIME` command will reveal the problem instantly.

Compatibility and Security

If the previous example with `SHOW_TIME` is served through NetCloak, the View Source or Show Source command in the browser will reveal that the NetCloak commands do not appear at all... only the message appropriate to the current time ("Good Morning", "Good Afternoon", "Good Evening" or "Good Night"). The NetCloak commands are never sent to the browser. This is because NetCloak is processing the `HIDE` and `SHOW` commands on the server, and the HTML is only sent to the browser after the commands have been processed.

Processing the commands on the server has two advantages: compatibility and security. Since only plain-vanilla HTML is ever sent to the browser, there is never any risk of the browser being incompatible with the NetCloak commands. Your dynamic pages will always work no matter what browser the user views them in. This is true for search engines and other non-human browsers, too.

When the commands are processed on the server, they are never sent to the browser. Users cannot "peek" inside your dynamic pages to see what is going on or find information or options that are hidden by NetCloak. This means that no-one will steal your NetCloak code or see something you are hiding with NetCloak when they view the source of the page.

Example: Hiding Internal Links

Consider a home page with a series of navigation links, like this:

```
<A HREF="/Contact.html">Contact Us</A>
<P>
<A HREF="/Help.HTML">Help Using This Site</A>
<P>
<A HREF="/InternalHome.html">For Employees Only</A>
<P>
```

In this example, the first two links on the page are intended for every user that comes to the page. The third link, however is for employees only. This page might provide a menu of local intranet applications,

QUICK START

databases, Web management features, etc. While the internal only page may be password protected, it is much more secure for outside users to never see the link to the internal site. Outside users won't be curious or confused by links that they are unable to follow and are not meant to access.

NetCloak can be used to hide this link from outside users by checking the client's TCP/IP address. If the local network TCP/IP address range is from 192.168.1.1 through 192.168.1.255, for example, the internal only link can be hidden like this:

```
<A HREF="/Contact.html">Contact Us</A>
<P>
<A HREF="/Help.HTML">Help Using This Site</A>
<P>
<HIDE><SHOW_DOMAIN BEGINS 192.168.1.>
<A HREF="/InternalHome.html">For Employees Only</A>
<P>
<SHOW>
```

You can see above that SHOW and HIDE commands, like INSERT commands, begin with the full command name and are followed by parameters that specify the conditions when the SHOW or HIDE command will be triggered. Usually the parameter list begins with a comparison operator, followed by one or more values to be compared.

The interesting addition is the fifth line, which starts with an unconditional HIDE command. This turns the text stream off for all users. Immediately following this command is a SHOW_DOMAIN command, which will turn the text stream back on for any user whose TCP/IP address BEGINS with "192.168.1.". In other words, the stream is turned back on only for users on the local network. The internal only link is then sent only to those users. When the HTML designed only for internal users is complete, a SHOW command turns the text stream back on for all users.

SHOW Never Hides, And HIDE Never Shows

An important point to note in the above example is that the SHOW_DOMAIN command is preceded by a HIDE command. If the text stream is already on when a NetCloak SHOW command is

encountered, then the result of the SHOW condition is irrelevant and the text stream will remain on after the command executes. To follow the water faucet analogy: if the faucet is on, it will remain on until it is turned off. If you want to have the faucet turned on only if a certain condition is true, you must first turn it off.

This behavior, at first, can lead to mistakes in creating conditional HTML. But ultimately it provides a great deal of flexibility in creating complex conditions. As you get started with NetCloak, just remember to precede conditional HIDE commands with `<SHOW>`, and conditional SHOW commands with `<HIDE>`. Thinking in terms of SHOW and HIDE may take a little getting used to. Keep the faucet analogy in mind and soon you will get more comfortable with it.

NetCloak Tip: The examples shown here are used to SHOW and HIDE various text messages, but NetCloak can be used to control any HTML element. The text stream that is turned on and off can contain text, formatting options, embedded images, hypertext links, or any other HTML content.

Example: Hide and Show with Counters

Counters can also be used to create conditional HTML. Here is an example that will display a special message to the first 100 people that come to your Web site:

```
<HIDE><SHOW_COUNT Winners LT 100>
Congratulations, you are lucky visitor number
<INSERT_COUNT Winners>!
<P>
<A HREF="winner.html">Click Here To Collect Your
Prize</A>
<SHOW>
```

Here, when the counter Winners is Less Than (LT) 100, the winner message will be displayed, the user will be told what visitor number they are, and a link will be shown to direct them to directions for claiming their prize. It is important to always use HIDE and SHOW COUNT commands along with an INSERT_COUNT command, since HIDE and SHOW COUNT will not increase the counter value. Without the INSERT_COUNT command, the "Winners" counter would never

QUICK START

increase and the winner text would always be displayed, even after the first 100 times the page is served.

Example: Identifying Users with NetCloak

Another common HIDE command is HIDE_USERNAME. This command can be used to present users with different information based on their access privileges. For example, let's say that both Bob and Tim have access to a Web page. While both users have access to the same Web page, a single item on that page can be hidden from Bob like this:

```
<SHOW><HIDE_USERNAME "Bob">
```

This message is for Tim only... Bob won't see it.

```
<SHOW>
```

Note that this is essentially the same as:

```
<HIDE><SHOW_USERNAME "Tim">
```

This message is for Tim only... Bob won't see it.

```
<SHOW>
```

In the first example, the message is hidden from Bob but displayed to everyone else. In the second, the message is shown only to Tim but hidden from everyone else. Consider this difference carefully when building pages with conditional commands.

If there are only these two users the different examples above function identically. This will probably not be the case, though. The logic of the first is “hide only from Bob” and logic of the second is “show only to Tim”. The first, as you can see, will be shown to everyone, not just Tim, but the second will be hidden from everyone except Tim.

Please note that the HIDE and SHOW_PASSWORD and HIDE and SHOW_USERNAME commands will not request the user to enter a username and password. They work based on what has already been entered. If you want to have the user's Web browser display a dialog box requesting the user to enter a password, you will need to use the NetCloak REQUEST_PASSWORD command, or some other security tool.

NetCloak PASSWORD commands will properly act on the user's password, regardless of what security tool required the password

initially. For example, if the user enters a password to gain access to a secure realm defined in your Web server software, the password will still be available to NetCloak for conditional processing.

Example: Tailoring Pages to Different Browsers

A constant challenge to web designers and developers is that different browsers on different platforms display HTML pages differently. For example, Windows browsers tend to display text larger than Mac browsers. Fortunately, NetCloak allows you to recognize the Web browser and create conditional HTML based on it. Here is the example:

```
<HIDE><SHOW_CLIENT CONTAINS "Win">  
<FONT SIZE=-2>  
<SHOW>
```

In this sample, the HTML FONT command will be sent to any client whose name includes "Win". NetCloak CLIENT commands are another excellent case where an INSERT command can be helpful. If you want to create conditional HTML based on the client type, use an INSERT_CLIENT command on a page, then load the page in several different browsers to see exactly how the browsers identify themselves.

By now, hopefully, the logic behind conditional HTML is starting to become clear. There are many more examples in the Reference Guide and on the NetCloak Examples Server at <http://examples.maxum.com/>

More Useful Commands

By now, you should be starting to get a feel for the kinds of things that NetCloak can do on your site. This page will introduce some more NetCloak commands, starting with a couple that that can be incredibly useful.

The REDIRECT Command

The REDIRECT command, as you would expect, allows you to redirect the client from a page on your site to another URL. This happens without the user clicking a link- the new URL just appears in their browser like magic. REDIRECT alone can solve a myriad of site

QUICK START

maintenance problems where published URLs must remain even though you have restructured the site.

Using the REDIRECT command is fairly simple. Just include the URL you wish to redirect the user to. The URL need not be on your own site- it can be any URL at all. For example:

```
<REDIRECT "http://www.maxum.com/ NetCloak/UsersGuide/">
```

REDIRECT has the same effect as a META REFRESH tag, but with two important differences. Like all NetCloak commands, it is processed on the server, and so it will work with all web browsers. And, as you can see above, the command is much simpler to remember- just REDIRECT and the URL in quotes.

The REDIRECT command can be even more powerful when combined with conditional SHOW and HIDE commands. Working together, these can redirect clients to a different page depending on their browser type, the URL they requested or any other SHOW/HIDE comparison. There are several examples of this technique on examples.maxum.com which you will see later.

The REQUEST_PASSWORD Command

Another useful command is REQUEST_PASSWORD, which allows you to require the user to enter a username and password to gain access to the page that includes the command. For example, to protect a page so that only the users Tim and Bob can access it, you could add this command to the page:

```
<REQUEST_PASSWORD "Realm" "Bob,Murphy" "Tim,Ross">
```

The first parameter (Realm) is sent to the browser as part of the password challenge. Most browsers will display this text in the password dialog box to give the user a note as to what page or service they are logging in to. For this parameter, you can use any text you like, and it will be passed through to the browser. The message usually looks something like this:

```
Enter a password for [realm]
```

After that, a list of username and password pairs tells NetCloak which users have access to the page.

Note that NetCloak will also track user accounts for you, allowing you to create several groups of users without having to include all of their usernames and passwords in the REQUEST_PASSWORD command itself. For complete details, see the command description in the Reference Guide.

ADVANCED TECHNIQUES

Now that the basics of using NetCloak's dynamic page commands have been covered, this section will continue with some more advanced techniques.

Cumulative Effects of Commands

The Introduction introduced the concept that the SHOW command will never cause a HIDE and vice-versa. For instance,

```
<SHOW_DOMAIN BEGINS 123.45>
```

will show text following it for those in the domain "123.45", but it will not hide text from those outside the domain. To do this, you must precede the SHOW command with an unconditional HIDE. To show only to clients in the domain 123.45, use this instead:

```
<HIDE><SHOW_DOMAIN BEGINS 123.45>
```

Think of the text of your document as a stream of text flowing through a faucet. When you use a conditional SHOW command, you are saying "turn the faucet on if this is true" but you are not saying "turn the faucet off if it is not true". The same rule applies to HIDE.

This allows using the commands in a cumulative way. The cumulative effect is best seen by an example:

```
<SHOW>
```

```
Here is the first line, visible to everyone.
```

```
<HIDE_DOMAIN IS some.domain>
```

```
This is hidden from users in "some.domain".
```

```
<HIDE_DOMAIN IS another.domain>
```

```
Hidden from "another.domain" AND "some.domain".
```

```
<SHOW>
```

```
This last line can be seen by everyone.
```

If we had wanted the sixth line to be hidden **ONLY** from "another.domain", the HTML would have been:

<SHOW>

Here is the first line, visible to everyone.

<HIDE_DOMAIN IS some.domain>

This is hidden from users on "some.domain".

<SHOW><HIDE_DOMAIN IS another.domain>

Hidden from "another.domain" **ONLY**.

<SHOW>

This last line can be seen by everyone.

The extra SHOW command turns the HTML on before the HIDE command turns it off for the desired users. This can get a little confusing, but there is a simple rule.

When you want to specify "Hide only from ...", use:

<SHOW><HIDE ...>

And when you want to specify "Show only to...", use:

<HIDE><SHOW ...>

This is very important, as the next example shows:

<SHOW>

This line is visible all the time!

<SHOW_DAY BEGINS SAT SUN>

This line is ALSO visible to the world.

In this example, since everyone could already view the HTML text, SHOW_DAY didn't do anything. To do what we want, which is to show the line only on the weekends, this example should be:

<SHOW>

This line is shown all the time!

<HIDE><SHOW_DAY BEGINS SAT SUN>

This line is visible **ONLY** on the weekend.

<SHOW>

By hiding first, we block access from everyone and then let people see the text **ONLY** when we want.

Why does NetCloak work like this?

Because, despite the fact that it takes a little getting used to, it gives you a lot more flexibility. For example, let's say you wanted to have a section of your document available only to your company all the time. The rest of the world could see the section during non-business hours only. In this case, you could do this:

ADVANCED TECHNIQUES

<SHOW>

This line is always available to everyone

<HIDE>

<SHOW_DOMAIN IS my.local.domain>

<SHOW_DAY BEGINS SAT SUN>

<SHOW_TIME GT= 19:00>

<SHOW_TIME LT 06:00>

This line is available to my local users all the time, and to the rest of the world during off hours.

<SHOW>

This example first hides the HTML text, then shows it for each specific case.

Using Negative Comparisons

As you've seen, you can show and hide portions of your pages based on a wide range of variables within NetCloak. This gives you a great deal of flexibility and you can create fairly complex conditions in which each part of a page will be visible. You can also negate the comparison in conditional SHOW and HIDE commands which allows you to show or hide in all cases **except** when the condition is true.

For example, assume you wanted to have a portion of your page visible during the normal business week, Monday through Friday. You could write:

<HIDE>

<SHOW_DAY BEGINS MON TUE WED THU FRI>

This text is visible during the week only.

But there is another way to make this same statement. Instead of saying "show text on Monday, Tuesday, Wednesday, Thursday, and Friday" (as we did above), you could say "show text on all days *except* Saturday and Sunday". To do this, you would use:

<HIDE><SHOW_DAY# BEGINS SAT SUN>

This text is visible during the week only.

In reality, this isn't really much easier, at least in this example. But what if you wanted to show text based on SHOW_DOMAIN to all users except those at a certain address. You certainly wouldn't want to list every domain on the Internet, so <SHOW_DOMAIN# 192.1.2.3>

could be very handy when you want to show a section of your page to everyone *except* those at the address 192.1.2.3.

Of course, you could, in many cases, also use the HIDE commands instead. For example, the “192.1.2.3” example could have been written as:

```
<SHOW><HIDE_DOMAIN 192.1.2.3>
```

The real power of negating comparisons to make “except” statements comes into play when combining commands.

Combining Commands by Negating

When building more complicated conditionals, using HIDE in place of SHOW except is no longer an option. Remember that SHOW and HIDE are fundamentally different in that SHOW will never “hide” text, and HIDE will never cause text to be seen. So, if we wanted to have a portion of text visible only to Joe from domain 192.1.x.x, we could **not** use:

```
<HIDE><SHOW_USERNAME IS JOE>  
<SHOW_DOMAIN BEGINS 192.1.>
```

This text would be visible to "JOE" AND to users coming from "192.1"!

In this case, as long as the user's name was Joe, it wouldn't matter what their domain was, because the `<SHOW_DOMAIN 192.1.>` command would never turn the faucet off. Joe's all over the Internet would have access to the text. Similarly, anyone in the domain 192.1 would also see the text, no matter what their name.

Instead, you could use the except variation to achieve the desired effect:

```
<HIDE><SHOW_USERNAME IS JOE>  
<HIDE_DOMAIN# BEGINS 192.1.>
```

This is what we wanted! Only Joe can see this, and only if he's at 192.1.

The first part of this is simple enough... Hide text, then let Joe see it again. The command `<HIDE_DOMAIN# BEGINS 192.1.>` (note the “#” character) is really saying “OK, text is on, so it's Joe. Now hide the text **except** when he is in the ‘192.1’ domain”.

Another common use for this technique is with username and password commands. These commands each work on the username

ADVANCED TECHNIQUES

or password, but not both. To show based on the username and password, use:

```
<HIDE><SHOW_USERNAME IS clarus>  
<HIDE_PASSWORD# IS moof>
```

The SHOW_USERNAME will only show HTML following it if the username is “clarus”, and the HIDE_PASSWORD# will hide it again except when the password is also “moof”. This results in the HTML following the show/hide-except commands being shown only when the username and password are both correct.

You’ve no doubt noticed by now that there is usually more than one way to achieve the same result, by converting SHOWs to HIDEs and then negating the comparison operator, or by using an “except” variation of a command, or by using a different operator. For instance, a “LT” operator is identical to a “#GT=” operator. You may use whichever combination of comparison operators and commands you feel most comfortable with that achieves the desired results.

Using Nested INSERT Commands

Finally, you can combine the power of NetCloak commands in more interesting ways, by nesting any INSERT command as a parameter inside another NetCloak command. When nesting commands, the nested INSERT command must not be enclosed in double-quotes (otherwise, it is treated as plain text). Here are some examples of nested commands:

Create a counter with the name of the current URL:

```
<INSERT_COUNT <INSERT_THISURL>>
```

Maintain a count of hits from each distinct browser type:

```
<INSERT_COUNT <INSERT_CLIENT>>
```

Redirect to a user-selected URL:

```
<REDIRECT <INSERT_VARIABLE selectedURL>>
```

Insert a macro file based on the current date

```
<SET_LOCAL todaysMacro = "/DatedMacros/" & <INSERT_DATE  
SHORT>>  
<MACRO <INSERT_LOCAL todaysMacro>>
```

Redirect to a selected page in a certain folder:

```
<REDIRECT /newPath/<INSERT_VARIABLE selectedPage>>
```

Using Nested INSERT commands can be tricky, and we recommend only using them when you are very familiar with how NetCloak works and the commands being used.

MACROS

A macro is a block of text or HTML that is defined in a single place which may then be used repeatedly in any of your cloaked documents. This gives you the ability to create a section of HTML text once and then use it on many different pages on your server. Changing the macro will change the text that is inserted everywhere the macro is used.

For example, many organizations put the same header and footer information on every page of their site. This helps to maintain a consistent look and feel and to make sure that navigational links to the home page, help, or other pages of interest are always available. Using macros, the header and footer can be defined once in a single place and then used on all pages. This makes it easier to create new pages and makes updating the header and footer quick and easy.

Macros can contain other macros, or can be used to include entire files inside of an HTML page.

There are two steps in using macros: defining the macros and then using them in your documents. Neither step is complicated.

Defining Macros

Begin by creating a file called NetCloak.macros in the NetCloak Files folder. This will be a standard text file, so use a text editor of your choice. Even SimpleText will do.

Once the file is open, make sure it is blank and you are ready to create a macro. Use the command **<BEGIN_MACRO macroName>** to start each macro, where macroName is the name of your macro. For example, if you want to create a standard header for your documents, you might start your macro with

```
<BEGIN_MACRO Header>
```

Next, enter the HTML text you would like to be inserted when the macro is processed. You may enter as much or as little text as you like. When your macro is complete, end it with the **<END_MACRO>** command.

An example should make this clear:

```
<BEGIN_MACRO NavigationLinks>
<HR>
<A HREF="http://192.1.2.3/Home.html">
Go To Our Home Page</A>
<A HREF="http://192.1.2.3/Help.html">
Get Help</A>
<HR>
<END_MACRO>
```

Whenever a cloaked document invokes the “NavigationLinks” macro, the text between the BEGIN_MACRO and END_MACRO commands will be inserted.

The NetCloak.macros file may contain as many macros as you like, but the total size of the file must be less than 64k.

Using Macros

Any cloaked document can use any macro defined in your NetCloak.macros file. Wherever you want to use a macro, simply use the command **<MACRO name>**, where “name” is the name of the macro. For example, to use the macro defined above, the cloaked document might end with this HTML text:

```
THANK YOU FOR VISITING OUR WEB SITE!
<MACRO NavigationLinks>
```

At the bottom of this page, the links to the home page and to the help system would be displayed, just as if they had been put directly into the document.

The macro command can also be used to include entire files, in addition to HTML snippets defined in the NetCloak.macros file. In place of the macro name, simply specify a file name, including the path, where appropriate. For example:

```
The file "Extra.html" looks like this:
<P>
<MACRO Extra.html>
```

In this case, the entire file Extra.html will be inserted into the page.

Inserting entire files can be useful when you don’t have enough space for a large macro in the NetCloak.macros file (the entire file is limited to

MACROS

64k) or when the file being inserted is generated by some other application.

The macro command will insert files located in the root folder hierarchy. If you would like to insert a file outside the root folder, make a Finder alias to the file and place the alias in the root folder or another folder contained within the root folder.

Note that if the Security setting ‘Prohibit file macros’ is enabled, you cannot use the MACRO command to insert the contents of files into your cloaked pages. See the section entitled “Security” for more information.

Changing Macros

Like NetCloak aliased documents, the macro file is loaded into RAM when NetCloak is started and must be reloaded whenever you make a change. If you add or change a macro, you must have NetCloak reload all the macros. You can quit your web server and restart it, but there are additional methods that are simpler.

1. Click the Reload Documents link in the HTML page of the Web administration interface.
2. Send the reload command from a web browser using the URL:

`http://your.server.com/RELOAD.nclk`

The RELOAD command URL must be used with a suffix that is mapped to NetCloak— .nclk will always work if your server has the NetCloak plug-in installed. The suffix is ignored, so any suffix that is mapped to NetCloak will work regardless of whether your server is using the plug-in or CGI version of NetCloak.

3. Choose Reload Documents from the File menu of the NetCloak CGI application. (This option is only available if you are using the NetCloak CGI.)
4. Click the Reload Documents button in the HTML tab of the Configuration window of the NetCloak CGI application. (This option is also only available in the CGI.)

You will note that the Reload command is the same one necessary to reload aliased documents cached in RAM. When you use Reload, both aliased documents and macros will be reloaded.

Cloaked File Limitations

Due to limitations in how CGIs interact with the server, the NetCloak CGI is limited to a file size of 60k bytes. All files larger than 60k will be truncated by the CGI before they are passed to the server. In some cases, even 60k may be too much data for your server to handle in a single request.

This limitation does not apply to the NetCloak plug-in. If you notice problems when serving large files (over 32k) with NetCloak, the best solution is to use the plug-in whenever possible. You can also read the NetCloak FAQ list at “www.maxum.com” for up-to-date information on handling larger files with the CGI.

NetCloak Caching and Aliases

In addition to dynamic page and forms-processing commands, NetCloak supports caching for HTML files and special NetCloak file aliases.

Dynamic Caching

NetCloak includes an intelligent dynamic file caching feature. Dynamic caching must be enabled using the Configuration page in the web admin interface, or the Configuration dialog in the NetCloak CGI. When enabled, NetCloak will cache the 100 most-requested files which fit into the available RAM. These 100 files are *in addition* to the files aliased (and cached) by the NetCloak.config file. Note, however, that NetCloak will only cache HTML files that it processes, not images or other files.

NetCloak’s dynamic cache checks the last modified date of cached files *every time the file is served*. Thus, when a cached file is modified, the new version will be automatically re-cached the next time it is accessed, without flushing other cached files. This ensures that files in the cache are always fresh, and eliminates the overhead of files being removed from the cache unnecessarily. The downside to operating this way is the small amount of time it takes to check the modification date each time a cached file is served.

SERVING CLOAKED FILES

NetCloak Aliases

NetCloak supports aliases to cloaked files in its configuration file, which resides in the NetCloak Files folder and is called NetCloak.config. A NetCloak alias is a shortcut that you can use in place of the full path to an HTML document. You should not confuse this with Finder aliases, which are unrelated.

Aliases provide several benefits, most important of which are:

1. Alias names add a layer of security by allowing you to hide the real name and path of the document from the user. Since these aliases only exist in NetCloak these are more secure than Finder aliases.
2. Aliased documents are automatically cached by NetCloak at startup and served from a very fast RAM-based cache.

The NetCloak.config file in your NetCloak Files folder is a text file that you can edit with any text editor or word processor. It consists of all of the NetCloak alias names and files that they correspond to. Each line of the file will contain an alias name followed by the path to the file to be served, separated by a `???` tab or a space.

All files placed in the NetCloak.config file will be read immediately into RAM when NetCloak starts. If you have frequently accessed pages on your server you may wish to place them in this file whether or not you wish to refer to them by another (aliased) name. The path may be:

- a **file name** - files in the root folder selected in NetCloak's Configuration window need only to be referenced by name.
- a **relative path** - files in a folder contained in the root folder selected in NetCloak should be referenced by a relative path name. Relative path names start with a colon or slash and include the names of each folder containing the folder where the file is located.
- a **full path** - files not contained in the root folder or a folder enclosed in it must be referenced by a full path. A full path must start with the name of the volume where the file is stored followed by the name of each folder containing the folder where the file is located.

For Example, the NetCloak.config file might look like this:

StartPoints.html	StartingPoints.html
RelativePath.html	:PageFolder:MyPage.html
FullPath.html	HardDisk:DocFolder:Full.html

Full or relative pathnames may use either colons or slashes to separate folder names.

Alias names may start with slashes, however, a leading slash will have no effect. NetCloak will not allow an alias to a folder, but if you wish, alias names may also include slashes to create the appearance that the files are coming from a particular folder. Links must contain the complete alias name, but any folder names included in the alias name need not actually exist.

There must be at least one space in between the alias name and the document path. Also, it is very important to end the alias name with the file extension used to cloak your files. If this extension is not included, the server will not use NetCloak to process the pages and will return a File not found error to the user.

To browse or link to an aliased document, simply reference the alias name as if it were in the root folder. For example, the file FullPath.html above would be linked by the URL:

`http://your.server.com/FullPath.html`

Making Changes to NetCloak Aliases and Cached Documents

As stated above, aliased documents are loaded into RAM when NetCloak is started. If you add or change a NetCloak aliased document, you must have NetCloak reload all the aliased documents. One way to do this is to quit NetCloak and restart, but two additional methods are provided.

1. Send the reload command from a web browser. Request the URL below (substituting the name or address of your server:

`http://your.server.com/RELOAD.nclk`

Actually, any URL with "RELOAD" in all caps and a suffix that will be processed by NetCloak will work.

2. Choose Reload Documents from the File menu within the NetCloak CGI. (This option is only available if you are using the

SERVING CLOAKED FILES

NetCloak CGI, and won't work if you are cloaking using the NetCloak plug-in.)

USING NETCLOAK WITH OTHER TOOLS

Sometimes it is useful to use NetCloak in conjunction with another server add-on tool to process a single request. For example, you might have a page that is generated by a CGI linked to a database, but would like to include NetCloak commands in that page. This is called “CGI Chaining”, and is made possible by NetCloak’s ability to call other plug-ins or CGI applications.

NetCloak can call other CGIs or plug-ins in one of two ways, using either the EXEC command, or by creating a suffix mapping so that requests for a specified file type are automatically routed through NetCloak to a “back-end” CGI.

The EXEC_CGI command is useful when you want to run more than one additional CGI for a particular page, or when you write a CGI that returns just a part of a page. Similarly, EXEC_PIXO can be used to insert the output of many third party plug-ins. For example, you could write an AppleScript CGI that generated some random phrase to be inserted into your HTML page. Instead of returning an entire page (HTTP header and all), your script could return only the phrase it was written to create. The results of your script would then be read and inserted into the page as the page is processed by NetCloak. Any NetCloak commands in the results from the CGI will also be processed by NetCloak. For specifics of using the EXEC_CGI and EXEC_PIXO commands see the Reference Guide.

In addition to executing a CGI or plug-in as simply one part of an entire page, NetCloak can also accept entire pages generated by other CGIs and parse the NetCloak commands that appear in the resulting page. For example, if you wanted to include NetCloak commands within a Lasso response page, you would set up Lasso as a “back-end” to NetCloak.

Calling other Plug-Ins via PIXO

When using the NetCloak plug-in, you may also make requests to other plug-ins that conform to the PIXO standard for inter-plug-in communication. In this case, the entry in the NetCloak.bsm file looks like this:

```
.NQRY TANGO_PIXO
```

where .NQRY is the file suffix that is mapped to the plug-in, and TANGO_PIXO is the name of the PIXO service registered by the plug-

USING NETCLOAK WITH OTHER TOOLS

in (in this example, by the Tango 2.x plug-in). Check with the vendor of your other WebSTAR API plug-ins to see if they support PIXO. If they do, they can provide you with the exact name of their PIXO service.

How Back-Ending Works

When a request comes in to the web server and is dispatched to NetCloak, NetCloak checks to see if the file should be handled by another CGI or plug-in. If so, NetCloak immediately passes the request off to the appropriate process. The back-end process handles the request normally, but instead of returning the response page to your web server, it returns the response page to NetCloak. NetCloak then receives the page, processes any NetCloak commands that appear in the page and returns the final result to the web server, which sends the response to the client.

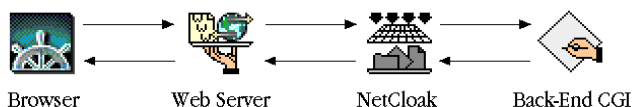


Figure 1: Flow of back-end CGI requests from the browser to the back-end CGI through NetCloak.

Defining Back-End Processes

To set up a back-end CGI or plug-in, you must:

1. Make sure that the action that defines how your web server is going to handle the request points to NetCloak. For example, Lasso files usually end in “.las”. To use Lasso as a back-end process, the suffix “.las” must use NetCloak for processing. Note that the web server will be sending these requests to NetCloak, not the CGI being back-ended! All CGI actions that will be cloaked must be referred directly from the web server to NetCloak.
2. Add a line to the NetCloak.bsm file. BSM stands for Back-end Suffix Mappings. This file tells NetCloak where to send requests that end with a particular suffix. It is made up of a file suffix (beginning with a period) and then the name of and path to the

USING NETCLOAK WITH OTHER TOOLS

CGI application or plug-in to be called. Here is an example .bsm file:

```
.Q      :Tango.cgi  
.RSP    :Responder.cgi
```

In this case, URLs with a suffix of .Q will be sent to Tango.cgi, and those ending in .RSP will be sent to Responder.cgi. The pathnames must be relative to the Root folder, and begin with a colon or slash.

Again, note that in your web server software, all of these suffix mappings should be directed to NetCloak so that NetCloak can hand off the request and then process the results.

A few additional notes:

When you give NetCloak the name of a CGI to execute in the .BSM file, the filename must be that of the actual application. The filename cannot be a file that is to be processed by an application or an alias to the CGI. You must tell NetCloak the full, exact name of the CGI application to be executed.

If the CGI is not in the same folder as NetCloak, you must specify a pathname for the CGI. Like aliases, pathnames in the NetCloak.bsm file must be paths relative to the root folder configured in NetCloak, and must start with a slash or colon. File and folder names in the pathname may be separated by slashes or colons.

You may use the .BSM file to specify up to thirty-two back-end processes.

Because of the extra overhead involved, back-ending CGIs can cause a noticeable decrease in performance on your server for pages served from a back-end CGI. We recommend that you only use back-end CGIs when needed. If another CGI does not require the use of NetCloak commands, don't set it up as a back-end CGI.

URLs that call CGIs by referring to them directly can be run through NetCloak without setting up a line in the .BSM file. Set your web server software to send all .CGI or .ACGI requests to NetCloak and it will automatically pass these URL requests to the specified CGI application and process the results.

Query or Path arguments specified in the URL will be preserved when NetCloak calls the back-end CGI, up to a maximum of 1024

USING NETCLOAK WITH OTHER TOOLS

characters. Post arguments are preserved up to a limit of 24Kb. This limit is the same as in WebSTAR.

If the error file setting of your web server is set to use a CGI, NetCloak will automatically route invalid requests to it and process any NetCloak commands in the page returned by the error CGI.

Troubleshooting Back-End Setups

If you have trouble setting up a CGI as a back-end CGI, make sure that the CGI executes correctly by itself first. Set up your web server to call it directly, without going through NetCloak. If that doesn't help solve the problem, try temporarily replacing the CGI with the "Responder.cgi" application. Responder is handy because it shows you exactly what is being sent to the CGI. Move the CGI you are trying to back-end and copy and rename Responder in its place. If Responder works as the back-end CGI, then chances are good that your set-up is correct. If Responder doesn't return a meaningful page, then go back and check your web server and ".BSM" settings. You can download Responder from the Maxum Web site at www.maxum.com.

You may also need to change settings in the CGI you are attempting to run as a back-end. Some CGIs require configuration changes to operate properly as a back-end to NetCloak. Tango is a notable example.

NetCloak Pro adds an additional set of commands for processing data from HTML forms. These are in addition to the dynamic page commands, and are only available in the NetCloak Pro. Throughout this section and the Reference Guide this is assumed even though NetCloak Pro may not be specifically mentioned.

HTML forms are relatively simple to create, especially using modern HTML authoring tools. NetCloak is what processes the input from these forms after users click the Submit button, allowing you to make use of HTML forms on your site. NetCloak can take form data and perform any of several actions:

- Save it as tab-delimited text
- Send it as formatted e-mail
- Update an existing page on your site
- Create a new page on your site
- Set user or local variables, counters, and cookies

For more complex systems, NetCloak can take several actions after a single form submission or conditionally perform some actions based on the content of the form.

This section will introduce some vocabulary for forms processing, refresh (or introduce you to) your knowledge of HTML form tags, then present tutorials to introduce forms processing concepts with some example systems.

The tutorials are included in HTML format with the NetCloak demo package in the NetCloak Introduction. If you have not already reviewed these with a demo, you can install them now and see them in action as you follow along in this Guide.

Before You Get Started

There are a few terms you need to know to make the most of this User Guide. First, you should be completely familiar with **HTML** (Hyper Text Markup Language) and specifically **HTML Forms**. The terms **Entry Form**, or simply **Form**, used in this guide refer to an HTML document that accepts input from users. The form can contain fixed-length text fields, scrolling text blocks, pop-up menus, radio buttons, check boxes and any other input type supported in HTML.

PROCESSING FORMS

This documentation will also refer to user-written **Articles**. An article is entered into an entry form, and is processed by NetCloak Pro into a formatted HTML document viewable by other users. The word “article” may not be entirely accurate for the system you are creating, but is used here to keep things simple. For example, if you were creating a system for entering recipes, each recipe would be considered an article.

Perhaps most importantly, you will need an **FDML** (Forms Definition Markup Language) template document. This is the file that contains one or more instructions used by NetCloak Pro to process the data entered by a user. Each FDML document describes to NetCloak Pro how to store, create, or edit HTML documents, send an e-mail message, or turn data into a text file.

The following tutorials are provided to help get you started creating your own NetCloak Pro-based interactive Web server. There are many examples included with this NetCloak Pro distribution, accessible using your Web browser, and the following examples are designed to help you understand the basic concepts of the more popular NetCloak Pro functions.

Forms Processing Basics

This section will cover the basics of processing forms with NetCloak. Once you master FDML templates, directives and insertion commands you can implement on-line surveys, feedback forms or even threaded discussion forums that users can access with just their web browser.

Understanding Forms Processing with NetCloak Pro

Forms processing with NetCloak Pro is controlled by special template files you create using NetCloak's forms processing commands. These files tell NetCloak what to do with the data from the HTML form. Similar to a mail merge file in a word processor, the template also defines what the created text file, web page or e-mail message will look like.

The diagram below shows an overview of how forms processing works. First, the user enters data into an HTML form. When the user clicks the submit button, their web browser sends the data from the form to the server. With the data the browser also sends the action

specified in the HTML form, which tells the server to use a NetCloak Pro FDML template to process the data.

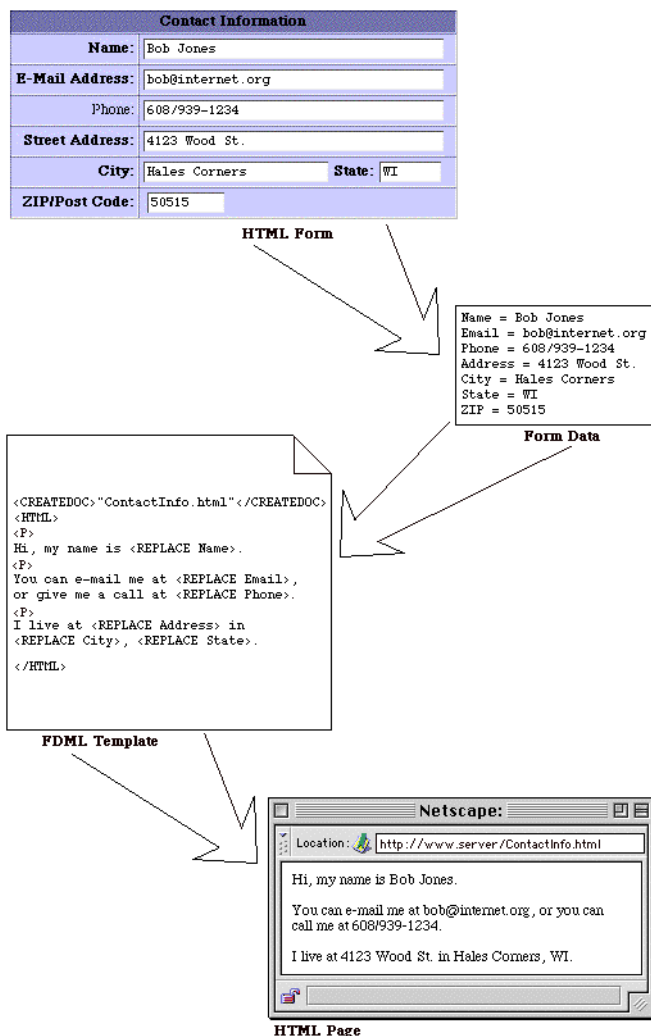


Figure 2: Overview of forms processing with FDML Templates

The server passes the data and the name of the template to NetCloak, which then reads the template. First in the template is a directive that tells NetCloak what to do- in this case, CREATEDOC tells NetCloak to create a new HTML document. The rest of the template determines what the contents of the new HTML page will look like, and NetCloak

PROCESSING FORMS

inserts data from the form in place of the REPLACE commands. At the bottom of the diagram is the new HTML page that NetCloak has created.

Creating HTML Input Forms

The first step in building all forms processing systems isn't the FDML file, but rather the HTML input form itself. All modern HTML editors will allow you to build HTML forms, and they can also be created using any text editor without much difficulty. Building HTML forms is outside the scope of this tutorial, and how this is done will vary depending on your HTML editor. See the user guide that came with your HTML editor, or an HTML reference for how to create input forms in HTML. The NetCloak Reference Guide also has a brief reference list of HTML form fields if you need to brush up.

There is one aspect of the form that is dictated by NetCloak, the FORM ACTION. Every HTML form includes a FORM tag, and one of the parameters of this tag, the ACTION parameter, tells the browser where to send the information once the user fills in the form and clicks the submit button. When you create the form, the ACTION must be the URL to the FDML template. The URL is no different from what would go in an HREF; it can be a full, root relative, or partial URL.

The FORM tag will look like one of these, depending on the URL you specify:

```
<FORM METHOD=post ACTION="Feedback.FDML">
<FORM METHOD=post ACTION="/Development/Test.FDML">
<FORM METHOD=post
ACTION="http://my.server.com/Support/Mail.FDML">
```

Form Processing Directives

The most important part of the FDML file is the directive which will tell NetCloak how to process the form data. There are four primary actions NetCloak can perform when a form is submitted, each of which is invoked using a different Primary Directive at the beginning of the FDML file:

1. **TEXTSTORE**: Stores the user input in a tab-delimited text file, suitable for importing later into a database, spreadsheet, custom application, text editor, etc.

2. SENDMAIL: Sends a formatted e-mail message to an Internet e-mail address.
3. CREATEDOC: Creates a new text file on the Web server. If these are HTML pages, there are supplemental directives that can link them to existing pages of the site.
4. INSERTFILE: Inserts new text or HTML into an existing file on the server.

In the overview diagram of forms processing (figure 2) you can see the CREATEDOC directive in the FDML template.

The tutorial examples following this one describe how to use the most common directives, SENDMAIL, INSERTFILE and CREATEDOC. The simplest, TEXTSTORE, is straightforward enough that you should be able to understand it with the explanation in the Reference Guide. One of the simpler and most popular of the remaining directives is SENDMAIL, so we will start there.

Before jumping right into how to make your own FDML template files, there are a few general points about FDMLs. To start, remember that FDML files are not HTML files, so the WYSIWYG HTML editor you are used to using is not a good choice for creating FDMLs. Some FDML files will create HTML or HTML documents, so an HTML editor can be useful for creating the design of the resulting pages. However, once you have roughed out the format with an HTML editor you will want to use a text editor like BBEdit to finish up. Using an FDML file to send an e-mail message, though, really has nothing to do with HTML so you should start and end with your text editor.

So why can't I use HTML Source mode in my HTML editor?

This may appear to work if you try it. When many WYSIWYG HTML editors output the final site, though, they will attempt to check and fix the HTML syntax. Since the NetCloak forms processing commands look like HTML, these applications frequently assume that they are malformed HTML tags and try to "fix" them. This "fix" will generally ruin your FDML template.

There is hope, though. Dreamweaver can be extended with objects for third-party add-on tags like NetCloak commands. A set of Dreamweaver objects for NetCloak is included in the package, and

PROCESSING FORMS

with them, you will be able to create pages with NetCloak's dynamic page and forms processing commands right in Dreamweaver. Similar extensions may be available for other editors soon.

Forms to E-Mail

This first example will be the simplest. Sending mail with NetCloak Pro is very straightforward using the SENDMAIL directive. This first example will cover the basics of sending mail, and also how to specify a custom response after the form is submitted.

Creating HTML Input Forms

The first step in building all forms processing systems is creating the HTML input form itself. To get started, fire up your favorite HTML editor and create a simple form for requesting information. On your form, create text fields called Name and Phone, and a text area called Message. Label them on the HTML page. When you are done, add a submit button. Don't worry about what it will do— we'll get to that later.

Every HTML form includes a FORM tag which includes an ACTION parameter. When you create any form to be processed by NetCloak, the ACTION must be the URL to the FDML file. For the form you just created, enter "Test.FDML", which is the name for the FDML file we will create in the next section. Don't add a leading slash or any folders or other path information. The FDML file will go in the same folder as the input form, and the browser should process the FORM ACTION correctly. Here is how your complete FORM command should look:

```
<FORM METHOD=POST ACTION="Test.FDML">
```

Starting the FDML File

When your HTML input form is ready, save it on the server like you would any other HTML page and open it in your Web browser. You should be able to fill in the various fields, but when you click the submit button, the server will return an error. You are now ready to create the FDML file, which will tell NetCloak how to process the form data.

Using a text editor (not your HTML editor), create a new file. This will be the FDML template that tells NetCloak to send data from the form

you just created as a mail message. Start with the SENDMAIL directive, which will specify what mail server to use to send the message, the address the message is going to, the address the message is from, and what the subject is. It should look something like this:

```
<SENDMAIL "mail.myorg.com" "bob@myorg.com"  
"tim@myorg.com">Form Submission</SENDMAIL>
```

Yours will be different from this because you will substitute your own addresses instead of those that are here.

The first parameter (mail.myorg.com in the sample above) should be the address or domain name of your local SMTP mail server. If you aren't sure of the name or address of your mail server, check the mail server setting of your own e-mail program to find out what it is.

The second parameter is the e-mail address of the recipient of the message. For now, simply enter your own e-mail address. This is followed by the From address, which is the mail account the message will appear to have been sent from. In the sample above, this parameter is set to tim@myorg.com, but for now set it to your own e-mail address as well.

NetCloak Tip: One of the most common problems encountered when using NetCloak to send e-mail is an invalid from address. Many, if not most, mail servers are configured to only process mail from known senders, which prevents spammers from using the mail server to forward junk mail. Unfortunately, this restriction also prevents legitimate mail forwarding. So, you need to be certain that the from mail address is a valid address and that the user specified has an account on your mail server. When in doubt, start by entering your own mail address for this field. Once you have confirmed that mail is being sent properly you can try changing it to another address.

Between the opening and closing SENDMAIL tags is the message subject. You can set this to anything you like, but for now make sure that the subject will be unique and easily identifiable.

The rest of the FDML file is simply the text of the mail message. For now, just type a line of text or two that will let you recognize the message. When you are done, your complete FDML file will look something like this:

PROCESSING FORMS

```
<SENDMAIL "mail.myorg.com" "bob@myorg.com"
"tim@myorg.com">Form Submission</SENDMAIL>
```

The form was submitted and NetCloak sent you this message.

Now save the file in the same folder on your Web server as the input form you created earlier, and name it "Test.FDML".

Testing the System

When that's done, go back to a Web browser and open the input form again through the Web server. Enter some information into the input fields, if you like, and click submit. This time, instead of an error message, you should get the generic NetCloak "Thank You" response. Wait a couple of minutes and check your e-mail account... There should be a new message waiting for you.

Debugging SENDMAIL

If the FDML example above doesn't work, you'll need to figure out why. The first possibility is that the server or NetCloak couldn't find the FDML file. If you get a file not found (404) error from the server, it did not send the request to NetCloak and the FDML file was not correct. In this case, make sure NetCloak is working properly and that the FDML is specified correctly and ends with a suffix that will be processed by NetCloak with the "NFORMS" action.

If you submitted the HTML input form and got a "Can't Find FDML File" error instead of the "Thank You" message, then NetCloak received the request but could not find the specified FDML file. Check to be sure that the FDML file is in the same folder as the input form, and check the FORM ACTION parameter to make sure it specifies the exact name of the FDML file you created.

If NetCloak did return the "Thank You" message but you don't receive the e-mail message after 5 or 10 minutes, then an error occurred sending the e-mail. Open the NetCloak Files folder on the server, then open the Failed Mail folder inside it. A text file should be in this folder; open it with SimpleText or BBEdit. It will include all of the details about the message, including the To and From addresses, the message body itself, and an error line that describes what went wrong. If the error message doesn't lead you to a resolution to the problem, send

the failed mail file to support@maxum.com and Maxum's support people will help you identify the trouble.

Extending The Example

Now that NetCloak has accepted the form submission and sent you e-mail, we have a good start. The trouble is that the e-mail message you get doesn't have anything interesting in it. That's easy to fix.

Remember that an FDML is just a template. Whatever message you included in your FDML file after the SENDMAIL command is what shows up in your e-mail message, just the way it appears in the FDML. If you put a blank line between two lines in the FDML file, then a blank line will be included in the e-mail message when you receive it.

Much more useful is to include information from the form in the message. To include any of this in the message, use the NetCloak REPLACE command to insert the field into the e-mail message according to the FDML template.

Your input form has the fields Name, Phone, and Message. To include this information that was entered into the form, add to the FDML file so it looks something like this:

```
<SENDMAIL "mail.myorg.com" "bob@myorg.com"
"tim@myorg.com">Form Submission</SENDMAIL>
A Message was submitted by <REPLACE Name>
(Phone: <REPLACE Phone>).
Message - <REPLACE Message>
```

If a user submitted the message form after entering the name John, the phone number 800.813.3410 and a short message, you would then receive an e-mail message like this:

```
A Message was submitted by John (Phone: 800.813.3410).
Message - This text was entered into the Message field on
the input form.
```

Other commands can also be used to represent the user entered information in an FDML template. In particular, the IF command is very useful. There is more about the IF command in the Guestbook tutorial later.

PROCESSING FORMS

The new form-to-e-mail system can be extended in other ways as well. In almost all cases you will want to return your own message when a form is submitted instead of the simple "Thank You" message built into NetCloak. This is done with the RESPONSE directive, which is put at the top of your FDML file, along with the primary directive. Here is how the FDML above would be changed to return the page "Response.html":

```
<RESPONSE>"Response.html"</RESPONSE>
<SENDMAIL "mail.myorg.com" "bob@myorg.com"
"tim@myorg.com">Form Submission</SENDMAIL>
A Message was submitted by <REPLACE Name>
(Phone: <REPLACE Phone>).
Message - <REPLACE Message>
```

Other directives allow your FDML file to confirm input of required fields, chain to other FDML files to perform additional actions, and more. The other tutorials will explain some additional techniques; for more, read the NetCloak Reference Guide.

Guestbook Tutorial: Adding to Existing Pages

In the previous tutorial you learned how to send mail from a form submission. The next step is to use forms to let users interact directly with your site by adding to an existing HTML page. We will build a simple Guestbook where users can add their name, address and a short comment.

The Guestbook example demonstrates the use of two important NetCloak Pro features. The INSERTFILE directive is used to update an existing HTML document, and the IF command is used to conditionally replace data from fields in the new HTML. It is also a good example of taking data from a user entry and rendering it as highly formatted HTML with IF commands to prevent the creation of broken or malformed links.

The Guestbook consists of three files- AddGuest.html, the HTML form where guests enter information; AddGuest.FDML, which processes the form and adds the new HTML to the third file, Guestbook.html. The file Guestbook.html is the actual guest book Web page.

The initial Guestbook.htm, with no entries, is very simple:

```
<HTML>
```

```
<HEAD>
<TITLE>Sample Guestbook</TITLE>
</HEAD>
<BODY>
Here is a registry of the people who have visited my
site...
Why not <A HREF="AddGuest.html">add yourself</A>?
<HR>
<!--GUESTLIST-->
</DL>
</BODY>
</HTML>
```

The unique feature of the file is the `<!--GUESTLIST-->` comment tag near the end. This comment serves as a marker where NetCloak Pro will insert new HTML into the document as users submit the form to add themselves to the page.

The entry form in AddGuest.html looks fairly long here, but is otherwise straightforward.

```
<HTML>
<TITLE>Add Yourself To The Guestbook</TITLE>
<H1>Add Yourself To The Guestbook</H1>
<FORM method=POST action= "/Guestbook/AddGuest.FDML">
To be added to the Guestbook, just fill out this form...
<P>
<HR>
<H3>About You...</H3>
<DL>
<DD><B>Name:</B>
<INPUT TYPE="text" NAME="UserName" SIZE="32"
MAXLENGTH="32">
<DD><B>E-Mail:</B>
<INPUT TYPE="text" NAME="EMail" SIZE="32" MAXLENGTH="32">
<DD><B>Title:</B>
<INPUT TYPE="text" NAME="Title" SIZE="32" MAXLENGTH="32">
<DD><B>Organization:</B>
<INPUT TYPE="text" NAME="Company" SIZE="32" MAXLENGTH="80">
<DD><B>Home Page URL:</B>
<INPUT TYPE="text" NAME="HomePageURL" SIZE="32"
MAXLENGTH="80">
</DL>
<HR>
```

PROCESSING FORMS

```
<H3>About Your Web Site...</H3>
<DL>
<DD><B>Site Name:</B>
<INPUT TYPE="text" NAME="SiteName" SIZE="32"
MAXLENGTH="40">
<DD><B>Main URL:</B>
<INPUT TYPE="text" NAME="MainURL" SIZE="32" MAXLENGTH="32">
<DD><B>Brief Description:</B><BR>
<INPUT TYPE="text" NAME="Desc1" SIZE="60" MAXLENGTH="60">
<BR>
<INPUT TYPE="text" NAME="Desc2" SIZE="60" MAXLENGTH="60">
</DL>
<P><HR><P>
<CENTER>
<INPUT TYPE=submit VALUE="Add Yourself">
</CENTER>
<P>
</FORM>
</HTML>
```

The form defines a number of fields for the user's name, e-mail address, company name, personal URL, company URL, and other interesting information.

This comes together in the AddGuest.FDML where the raw information is put into formatted HTML. It is a relatively short file, but a lot happens in it.

- 1) <RESPONSE>"/Guestbook/Guestbook.html" </RESPONSE>
- 2) <INSERTFILE "<!--GUESTLIST-->"> "/Guestbook/Guestbook.html"
</INSERTFILE>
- 3) <P>
- 4) <IF HomePageURL IS# "" THEN "<A HREF=<REPLACE HomePageURL>">
- 5) <REPLACE UserName>
- 6) <IF HomePageURL IS# "" THEN "">
- 7) <IF EMail IS# "" THEN "(<REPLACE EMail>)">
- 8) - <REPLACE Title>
- 9) <IF Title IS# "" THEN ", ">
- 10) <REPLACE Company><P>
- 11) <DL>
- 12) <DD><IF MainURL IS# "" THEN "<REPLACE MainURL>"">">
- 13) <REPLACE SiteName>
- 14) <IF MainURL IS# "" THEN "">
- 15) <IF Desc1 IS# "" THEN " - <REPLACE Desc1> <REPLACE Desc2>">

16) </DL>

17) <HR>

So, what's going to happen here after the user clicks the Submit button?

Line 1 - The RESPONSE directive is used in the other examples, but in this case note that the response file is the same Guestbook.html file that NetCloak Pro is updating!

Line 2 - The INSERTFILE directive contains the <!--GUESTLIST--> marker comment to indicate where NetCloak Pro should insert the new HTML in the page being updated. This must match the comment in the guestbook file. Between the opening and closing tags is the path and filename for the Guestbook.html file where the new HTML will be inserted.

Line 4 - This line creates a link to the user's home page if their URL. If the user did not enter a home page URL, the IF commands will prevent NetCloak Pro from creating a broken link.

Note that this requires two IF statements— one for the first part of the anchor tag, with "< A HREF =" and the URL of the user's home page, and another for the closing "< /A > " from the anchor tag (on line 6).

The IF commands are easier to understand if you look at each part separately:

```
<IF HomePageURL IS# ""
```

The first half of the IF command has a logical condition; the command uses the # character to specify "not equal to", so the snippet above has the logic "If the field HomePageURL is not empty". This is a common technique to check for empty fields in the user's input. The rest of the IF command inserts the URL into the HREF part of an anchor tag.

```
THEN "<A HREF=<REPLACE HomePageURL>>">
```

The second part of the IF command, following THEN, is text that will be inserted if the condition is met.

Here the second part of the IF command actually has a REPLACE command inside it. By putting the REPLACE inside the IF command, the replacement may not be made depending on the IF command. This works because NetCloak Pro executes all of the IF commands

PROCESSING FORMS

first, then goes through and executes the REPLACE commands. You can also use this technique to create very complex forums by conditionally executing RESPONSE or CHAIN directives.

Line 7 - The next line uses a similar IF command sequence to insert the user's e-mail address only if they have entered something into the EMail field.

Line 9 - The IF command here will insert a comma between the title and company name if the user has entered a title.

Line 11 - The <DL> tag will format the next few lines as a definition list so they will be nicely indented.

Line 12 - This line starts out with a <DD> tag to indent the HTML. Here is yet another IF command to create a link to the user's company Web site only if they have entered the URL for one.

Line 15 - This assumes that if the user didn't enter anything on the first description line, they probably didn't enter anything on the second either. If there is something on the first description line, it inserts both.

Lines 16-17 - These two close the definition list and then insert a horizontal rule to separate the new entry from the existing ones.

Recipes Tutorial: Creating New Pages

As an example of how to create new pages on your site with NetCloak Pro, we will build a very simple system for users to share recipes. This tutorial example uses the CREATEDOC directive to create new pages from information submitted by users, as well as the MENUDOC directive to automatically build a list of links to the new pages.

The exact same methodology could be applied to a system for posting bulletins, news articles, student essays, personal home pages, discussion forums, or any other system where users of your Web site need to be able to create content for other users. Turn your Web site into more than a collection of Web pages... make it an interactive resource for your users.

A Recipe System

The recipe system will have three main components: a menu, recipe pages, and a form where users can add recipes to the system. We'll

start with the recipe menu, which will look like this when just a few recipes have been entered.

As you can see, we'll be using very simplistic HTML formatting, to make reading the example as easy as possible.

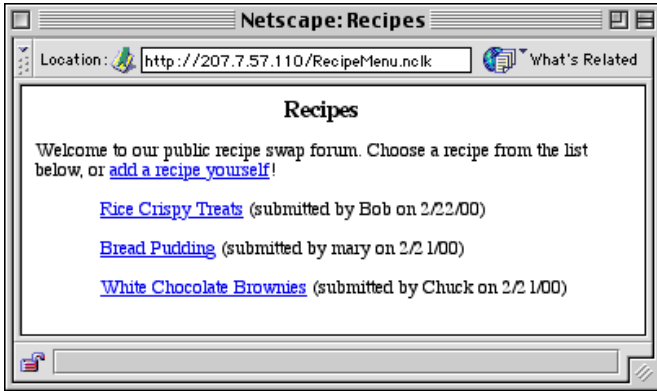


Figure 3: A sample Recipes list.

When a user clicks on one of the recipe links, they will be taken to a full recipe page. We'll prototype this page as well. Here is how the Rice Crispies page might look:

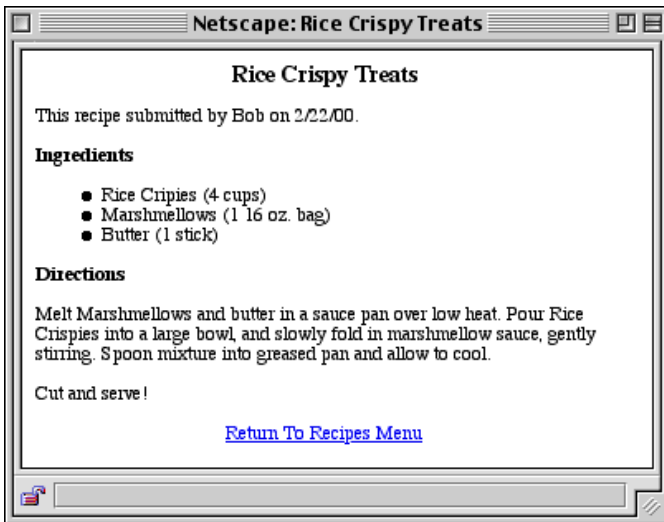


Figure 4: A sample recipe— Rice Crispy Treats

PROCESSING FORMS

Notice that we have created sample pages that can be used as prototypes for building the fully interactive system. You might be tempted to jump right in and build a highly interactive system in a single step, but it is always better to slow down and take things one step at a time. In the final system, the menu page will be automatically maintained and the recipes automatically created, but for now, we hard code the samples.

One more page will be needed, the input form that allows visitors to submit new recipes. Here is what that page will look like:

A screenshot of a Netscape browser window titled "Netscape: Recipes". The main content area is titled "Recipe Entry". It contains four labeled input fields: "Author:" with a single-line text box, "Recipe Name:" with a single-line text box, "Ingredients: (enter each ingredient on a separate line)" with a multi-line text area and a vertical scrollbar, and "Directions:" with a multi-line text area and a vertical scrollbar. At the bottom of the form area is a "Submit Recipe" button and a blue underlined link that says "Return To Recipes Menu". The browser's status bar at the bottom shows a small icon on the left and a progress indicator on the right.

Figure 5: The Recipe Entry form.

With this basic three page system, users can review the list of recipes, read (and print out) full recipe descriptions, and submit new recipes to the system. We could specify many additional bells and whistles, but again, the key is to start with a solid, properly working mechanism onto which new features can be added later.

Building The System

Each of the three pages shown above could have been created using any HTML editor or text processor. Let's take a look at the prototype recipe menu page...

```
<HTML>
<HEAD>
<TITLE>Recipes</TITLE>
</HEAD>
<BODY BGCOLOR=#FFFFFF>
<CENTER><H3>Recipes</H3></CENTER>
Welcome to our public recipe swap forum. Choose a recipe
from the list below, or <A HREF="EnterRecipe.html">add a
recipe yourself</A>!
<DL>
<!--RecipesGoHere-->
<DD><A HREF="RiceCrispyTreats.html">Rice Crispy
Treats</A>
(submitted by Bob on 5/22/00)<P>
<DD><A HREF="BreadPudding.html">Bread Pudding</A>
(submitted by Mary on 5/21/00) <P>
<DD><A HREF="WhiteChocolateBrownies.html">White Chocolate
Brownies</A> (submitted by Chuck on 5/21/00)<P>
</DL>
</BODY>
</HTML>
```

The entire page is made up of plain HTML. Note the link to the form, EnterRecipe.html.

The third section is the list of recipes, formatted using the definition list (DL) tag. Of note here is the HTML comment `<!--RecipesGoHere-->` which will be used by the FDML file later to specify the place in the file where new recipe listings should be added.

When a user decides to add a recipe, they will click on the "add a recipe yourself" link on the page and be taken to the input form shown above. The HTML source looks like this:

```
<HTML>
<HEAD>
<TITLE>Recipes</TITLE>
</HEAD>
```

PROCESSING FORMS

```
<BODY BgColor=#FFFFFF>
<CENTER><H3>Recipe Entry</H3></CENTER>
<FORM METHOD=POST ACTION="Recipes/Recipe.fdm1">
<B>Author:</B> <BR>
<INPUT TYPE="text" NAME="Author" SIZE="30"
MAXLENGTH="30">
<P>
<B>Recipe Name:</B> <BR>
<INPUT TYPE="text" NAME="RecipeName" SIZE="30"
MAXLENGTH="30">
<P>
<B>Ingredients:</B> (enter each ingredient on a separate
line)<BR>
<TEXTAREA NAME="Ingredients" ROWS=3 COLS=56></TEXTAREA>
<P>
<B>Directions:</B><BR>
<TEXTAREA NAME="Directions" ROWS=3 COLS=56></TEXTAREA>
<P>
<CENTER>
<INPUT TYPE="submit" VALUE="Submit Recipe">
</CENTER>
<P>
</FORM>
<CENTER><A HREF="Recipes/RecipeMenu.html">Return To
Recipes Menu</A></CENTER>
</BODY>
</HTML>
```

The form in the middle section, set off by `<FORM>` and `</FORM>` tags, is the interesting part. First, the `FORM` tag itself includes two crucial parameters. The first is `METHOD=POST`, which tells the browser to use the HTTP POST method for sending the data to the server. The POST method sends the form data to the server in part of the HTTP request that is hidden from the user, and allows a large amount of data to be sent. The other possible method is GET where the form data is sent in the URL. The GET method is limited by the maximum size of a URL. NetCloak does support the GET method, but POST is the best choice for most systems.

After the `METHOD`, the `FORM ACTION` is set. This is simply a URL to the FDM1 file that we will create below. The `ACTION` URL can be a file-relative partial URL, a root-relative URL, or a fully qualified URL

(beginning "http://"). In this case, a file relative URL is used since the FDML is in the same folder as the input form.

Following the FORM command are the four input fields we will need the user to complete. In this example, we specify two fixed length text fields, for the author and the recipe name. There are two scrolling text areas for the ingredients and directions because these will usually be several lines of text.

When the user completes the form and clicks submit, the FDML file is invoked and the real work is finally done.

Processing The Form Data

To create the FDML file, we start with the sample recipe we already created:

```
<HTML>
<HEAD>
<TITLE>Rice Crispy Treats</TITLE>
</HEAD>
<BODY BgColor=#FFFFFF>
<CENTER><H3>Rice Crispy Treats</H3></CENTER>
This recipe submitted by Bob on 5/22/00.
<P>
<B>Ingredients</B>
<P>
<UL>
<LI>Rice Crispies (4 cups)
<LI>Marshmallows (1 16 oz. bag)
<LI>Butter (1 stick)
</UL>
<P>
<B>Directions</B>
<P>
Melt marshmallows and butter in a sauce pan over low
heat. Pour Rice Crispies into a large bowl, and slowly
fold in marshmallow sauce, gently stirring. Spoon mixture
into greased pan and allow to cool.
<P>
Cut and serve!
<P>
<CENTER><A HREF="Recipes/RecipeMenu.html">Return To
```

PROCESSING FORMS

```
Recipes Menu</A></CENTER>
</BODY>
</HTML>
```

First, we'll remove the Rice Crispy specific text and replace it with NetCloak REPLACE commands. The REPLACE command is used to insert the contents of a field, as entered by the user, at the specified point. For example, the author's name (Bob) in the listing above will be replaced with a `<REPLACE Author>` command. The variable Author is the name of one of the input fields, and comes from the input form.

The template for the recipe page, with NetCloak commands used in place of the sample text, is:

```
<HTML>
<HEAD>
<TITLE><REPLACE RecipeName></TITLE>
</HEAD>
<BODY BgColor=#FFFFFF>
<CENTER><H3>REPLACE RecipeName</H3></CENTER>
This recipe submitted by <REPLACE Author> on <DATE>.
<P>
<B>Ingredients</B>
<P>
<BULLET Ingredients>
<P>
<B>Directions</B>
<P>
<REPLACE Directions>
<P>
<CENTER><A HREF="Recipes/RecipeMenu.html">Return To
Recipes
Menu</A></CENTER>
</BODY>
</HTML>
```

Notice that the `<REPLACE RecipeName>` is used twice, first in the page title and again as the headline of the page. This is perfectly legal, and form variables may be used as many times as you like in any FDMl file.

The line "this recipe submitted by" includes the `<REPLACE Author>` command to insert the author's name, followed closely by the

NetCloak `<DATE>` command. This tag, used without parameters, simply inserts the current date into the newly created file.

The Ingredients section includes another new command, `BULLET`. This command is similar to `REPLACE`, inserting the contents of the specified input field (in this case, the Ingredients field). `BULLET`, however, inserts data from the form field with HTML tags to make each line of the input field an item in an unordered list.

The directions are inserted using a plain `REPLACE` command. Of note here is that the text entered by the user for the directions may include carriage returns. NetCloak automatically converts a single return to an HTML `<BR>` tag and two consecutive returns to a `<P>` tag. This maintains the line and paragraph breaks when the directions are inserted into an HTML file.

Now it is time to tell NetCloak what to do with our template. A `CREATEDOC` directive will create the new page for the recipe, and a `MENUDOC` to add a link to the new page in the recipe menu. Finally, a `RESPONSE` directive will send our custom response page to the user after they submit the form instead of NetCloak's built in "Thank You" message.

Here is the directive block for the recipe system:

```
<CREATEDOC>"<REPLACE_FN RecipeName>.html" </CREATEDOC>
<MENUDOC "<!--RecipesGoHere-->">
"Recipes/RecipeMenu.html"
<DD><A HREF="<HTMLFILENAME>">
<REPLACE RecipeName></A>
(submitted by <REPLACE Author> on <DATE>)
<P>
</MENUDOC>
<RESPONSE>"Recipes/ThankYou.html" </RESPONSE>
```

All FDML files must include exactly one Primary Directive; in this case, `CREATEDOC`. This directive tells NetCloak to create a new HTML page on the server. Inside of the `<CREATEDOC>` and `</CREATEDOC>` tags, the name of the file to be created is specified.

The filename specified in the `CREATEDOC` command can be either just a filename (in which case the file created will be placed into the same folder as the FDML file) or a full root-relative path to place the new file in another folder. Paths specified inside FDML tags may not

PROCESSING FORMS

be full URLs. NetCloak processes files using the file system on the server, not HTTP, so it cannot create files on another server or site.

Also note the REPLACE commands inside the CREATEDOC directive to give the new file the same name as the recipe. A special version of the REPLACE command, REPLACE_FN, is provided specifically for use in filenames. The REPLACE_FN command works like the normal replace command, but also removes characters that may result in invalid file names or URLs. For example, spaces, which must be encoded in URLs, are removed altogether, so that when a user creates a recipe with the title Rice Crispy Treats the REPLACE_FN tag will insert RiceCrispyTreats. The resulting CREATEDOC command will then be:

```
<CREATEDOC>"Recipes/RiceCrispyTreats.html" </CREATEDOC>
```

The next directive in the block is the MENUDOC command. It tells NetCloak that a menu is being maintained, which will provide a link to the newly created page. Inside the opening tag of the MENUDOC directive is the HTML comment in the menu page which specifies where new links should be added. In the listing of the HTML source for the menu above, you can see that the comment `<!-- RecipesGoHere-->` appears at the top of the list, right where links to newly created recipes should go.

Immediately after the opening MENUDOC tag is the filename of the menu to be updated, enclosed by double quotes. Again, this is specified as a root-relative path, although a filename is also sufficient if the menu page is in the same folder as the FDML file.

After the filename of the menu page is the text and HTML that will be added to the menu. Here, we've taken a line from the menu and replaced the static sample text with NetCloak REPLACE, DATE, and HTMLFILENAME commands, just like making the recipe page template. Specifying the full text of the menu item to be inserted gives you the flexibility to format menu pages any way you like. In this case, the menu is formatted as a definition list, and not only provides a link to the page, but lists the author and the date the recipe was added as well.

The HTMLFILENAME command provides an easy way to link to the new page created by a CREATEDOC directive. This command will

insert a root-relative path to the new file, and is most frequently used to create a link to the new page inside the `MENUDOC` command.

The last FDML directive in the block is the `RESPONSE` command. This directive simply tells NetCloak what file to return to the user when the input form is submitted. Normally, this will be a very simple page thanking the user for their submission and providing a link or two back to key pages in the system. In this case, for example, a link would be provided to the recipe menu page. When using this directive in your own systems, remember it accepts the path to a file on the server, not a URL.

Creating This System Yourself

The system described here makes an excellent starting point for many other similar solutions, and is also a good base upon which to experiment with other NetCloak forms processing systems. To get the recipe system up and running on your own server, copy the Recipes folder out of the Extras folder included in the NetCloak package into your web server or site's root folder.

Now you are ready to try the system. On any Web browser, go to the recipe menu with the URL:

`http://your.server/Recipes/RecipeMenu.html`

Of course, you will need to replace "your.server" with the address or domain name of your Web server.

The menu page that appears will be quite short, because no recipes have been entered yet. For now, just click the "add a recipe..." link.

When the recipe entry page appears, fill out the form completely. To see that the system is working, complete the fields as you would if you were entering a real recipe. In particular, enter several ingredients, with each on its own line, and enter a few paragraphs of directions. When you are done, click the Submit button.

Notice that the next page displayed is the Thank You page. Return to the recipe menu by clicking the link on the page. Back on the recipe menu page, click Reload if the new recipe is not listed. If the recipe is still not shown, try a power reload by holding down the option key while clicking the reload button (if you are using Netscape Navigator).

PROCESSING FORMS

The new recipe should now appear on the menu, and clicking on the link should display the full recipe. Go back to the Finder on the Web server and take a look at the Recipes folder... a new file has been added and is named according to the recipe name entered. You can open and edit this file just like any other HTML page on your server.

Creating Pages: The Feedback Example

This example is a feedback that allows users to enter comments about the Internet into your server. Other users can then add responses to the original comments. Before you start reading how the example is put together, make sure you have entered a few articles and know how it works. Go to the NetCloak Introduction pages and click on Feedback under the Examples.

The Feedback home page has links to the original article entry form and to several separate menu documents. In this example, new articles will be broken down into one of several forums to make it easier for users to find what they want. This particular example uses the forums General Internet, This Site, and Other Sites, but it could be any other logical breakdown of topics appropriate for the system.

Each of the forums has its own menu document. These documents are all exactly the same, except for their titles and minor variations in their text. Most importantly, since each menu document will be updated by NetCloak Pro, each needs to have a single unordered list.

Most of the form should be self-explanatory. Note that the pop-up list for the topic name only lets the user select one of the pre-defined choices.

Now for the interesting part. The Feedback.FDML template will generate the original articles. Responses to original articles will be handled by a different FDML template. The Feedback.FDML file is as follows:

```
1) <CREATEDOC>
2) "/Feedback/<REPLACE_FN ForumName>Articles/
   <REPLACE_FN DocTitle>.HTML"
3) </CREATEDOC>
4) <AUTOLINK>"/AutoLink.doc"</AUTOLINK>
5) <MENUDOC>"/Feedback/<REPLACE_FN ForumName>Menu.html"
6) <LI> <A HREF="<HTMLFILENAME>"><REPLACE DocTitle>
   (<DATE>)</A>
7) </MENUDOC>
8) <RESPONSE>"/Feedback/Response.html"</RESPONSE>
9) <RECENTLIST>
10) <LI> <A HREF="<HTMLFILENAME>"><REPLACE DocTitle></A>
11) </RECENTLIST>
12) <HTML>
13) <HEAD>
14) <TITLE><REPLACE DocTitle></TITLE>
15) </HEAD>
16) <BODY>
17) <H1><REPLACE DocTitle></H1>
18) <ADDRESS>
19) This article submitted by <REPLACE Author> on <DATE>.
20) </ADDRESS>
21) <BR>
22) <REPLACE ArticleText>
23) <P>
24) <LINKNEXT><BR>
25) <LINKPREVIOUS><BR>
26) <A HREF="<MENUFILENAME>">Return to Topic Menu</A>
27) <HR>
28) <H3>Here is a list of responses that have been posted
29) to this article...</H3>
30) <UL>
31) </UL>
32) <HR>
33) <H3>If you would like to post a response to this
34) article, fill out this form completely </h3>
35) <FORM METHOD=POST
    ACTION="Feedback/FeedbackResponse.FDML">
36) Response Title:
37) <INPUT TYPE="text" NAME="DocTitle" SIZE="32"
    MAXLENGTH= "32"> <BR>
38) Author:
```

PROCESSING FORMS

```
39) <INPUT TYPE="text" NAME="Author" SIZE="32"
    MAXLENGTH="32"><P>
40) Response Text:<BR>
41) <TEXTAREA NAME="ArticleText" ROWS=10 COLS=60>
    </TEXTAREA><P>
42) Original Article:
43) <INPUT TYPE="text" NAME="OrigArticle" SIZE="20"
44) MAXLENGTH="40" VALUE="<REPLACE DocTitle>"
    (Don't change this field!)
45) <BR>
46) Forum:
47) <INPUT TYPE="text" NAME="OrigForum" SIZE="16"
48) MAXLENGTH="16" VALUE="<REPLACE ForumName>"
    (Don't change this field either!)
49) <P>
50) <INPUT TYPE="submit" VALUE="Submit Response">
51) <INPUT TYPE="reset" VALUE="Start Over"><p>
52) </FORM>
53) <HR>
54) Article complete. Click
    <A HREF="/Feedback/Feedback.html">HERE</A>
55) To return to the Feedback Menu.
56) </BODY>
57) </HTML>
```

Lines 1-3 - The CREATEDOC directive places the new document into a sub folder based on the name of the forum using the REPLACE_FN insertion command.

Placing documents into separate forum sub-folders is not necessary, but will help to keep the documents in each folder to a manageable number, in addition to providing a logical breakdown.

Notice the use of REPLACE_FN here where the value will be used as a file or folder name.

Line 4 - The AUTOLINK directive will create automatic links to any key words entered by the user which match entries in the AutoLink.doc file. See the section on the AutoLink directive for an explanation of how this works.

Line 5 - The MENUDOC directive again uses the REPLACE_FN command to update the menu document for the selected forum.

Line 6 - Note the use of the HTMLFILENAME insertion command. This command inserts the path and filename for the HTML file specified in the CREATEDOC directive. We could have repeated the parameter inside the CREATEDOC directive, but the HTMLFILENAME command is much shorter and makes it easier to understand what is going on.

Here comes the real trick to this example: The HTML documents generated by this FDML will contain not only the content of the original message, but an input form and menu of responses.

Lines 24-25 - The LINKNEXT and LINKPREVIOUS insertion commands here will automatically insert links to the previous article and next article entered. When it creates the HTML document for the current article, NetCloak Pro will put a placeholder in place of the LINKNEXT tag, and replace it with a real link when the next article is entered.

Line 26 - The MENUFILENAME here will automatically insert the complete name and path to the menu file specified in the MENUDOC directive. We could have repeated what was inside the MENUDOC directive, but this is a handy shortcut.

Lines 30-31 - Note the bullet list (, since it's empty to begin with), which is preceded by a text blurb stating that responses are available. This bullet list will be updated automatically when responses are posted to the original article.

Lines 35-52 - Here is another entry form at the end of the document which the FDML file will create in the new article. The last two fields (lines 43 and 47), when the form is displayed, will default to the correct values and should not be modified by the users. This is where the name of the original article is passed to the FDML which will process the response.

When a user fills in the response entry form, NetCloak Pro will again be invoked, but this time it will use the "FeedbackResponse.FDML" file:

```
1) <CREATEDOC>
2) "/Feedback/<REPLACE_FN OrigForum>Articles/
   <REPLACE_FN DocTitle>.html"
3) </CREATEDOC>
4) <AUTOLINK>"/AutoLink.doc"</AUTOLINK>
5) <MENUDOC>
6) "/Feedback/<REPLACE_FN OrigForum>Articles/
   <REPLACE_FN OrigArticle>.HTML"
```

PROCESSING FORMS

```
7) <LI> <A HREF="<HTMLFILENAME>"><REPLACE DocTitle>
    (<DATE>)</A>
8) </MENUDOC>
9) <RESPONSE>"/Feedback/Response.html"</RESPONSE>
10) <RECENTLIST>
11) <LI> <A HREF="<MENUFILENAME>">
    <REPLACE OrigArticle></A>
12) </RECENTLIST>
13) <HTML>
14) <HEAD>
15) <TITLE><REPLACE DocTitle></TITLE>
16) </HEAD>
17) <BODY>
18) <H1><REPLACE DocTitle></H1>
19) <ADDRESS>
20) This response submitted by <REPLACE Author> on <DATE>.
21) </ADDRESS>
22) <BR>
23) <REPLACE ArticleText>
24) <P>
25) <LINKNEXT><BR>
26) <LINKPREVIOUS><BR>
27) <A HREF="<MENUFILENAME>">Return to Main Article</A>
28) <HR>
29) Article complete. Click
    <A HREF=""/Feedback/Feedback.html">HERE</A>
30) To return to the Feedback Menu.
31) </BODY>
32) </HTML>
```

The notable thing in FeedbackResponse.FDML is that it uses the original article document as its menu document. By doing this, responses will be added directly in to the bullet list on the original article page.

One additional note about the Feedback example. Two fields in the "Feedback.FDML" file, "OrigArticle" and "OrigForum", were implemented as standard text fields in order to make what is happening more clear in the example. In a production system, you would most likely want to make these hidden fields so that the user won't ever see them. See an HTML language guide for more details on hidden fields.

Using the AUTOLINK directive you can create automatic links to your home page or sub-pages by any key words you choose. Automatic links can be established to company product pages, professional solutions, key employees, etc. Many of your organization's current intellectual assets in the form of text documents and graphics can be processed and published on your Web site with NetCloak Pro as well.

An AutoLink file is a plain text file created in any text editor or word processor. The file is made up of "Potential Links", or links that may be added to any article. There is one potential link on each line of the file. Each potential link is made up of a key word or phrase that should be linked along with an HTML command that will create the link. The keyword and the HTML command must be separated by one or more tab characters.

For example, a good start for an AutoLink file would be:

```
Maxum      <A HREF="http://www.maxum.com/">Maxum</A>  
Apple      <A HREF="http://www.apple.com/">Apple</A>
```

Any article generated from an FDML document that AutoLinks this file will have the words "Maxum" and "Apple" automatically replaced by hypertext-links to their respective home pages. You may, of course, include any links you want to documents on or off your own Web site. All the AUTOLINK directive does is replace the key words with the specified text.

AutoLink does not require that you replace words with hypertext links. AutoLink will replace any single word with another word or phrase. You could use AutoLink to replace the names of competitors with "Brand X" and "Brand Y", or replace offensive language with something more acceptable.

USING RESPONSE FILES

A Response file can be created to provide feedback to the author after they have entered an article. It might say:

Thank You!

Thank you for contributing to this server. You may now
Return to Our Home Page or Review Your Article.

The underlined portions would be linked to the appropriate places.
The text and format of the response file is entirely up to you.

In the earlier RESPONSE directive example, we used the example:

```
<RESPONSE>" /Feedback/Response .html"</RESPONSE>
```

This command points to response file "/Feedback/Response.html."
The Response file for the above example would look like this:

```
<HTML>
<HEAD>
<TITLE>Thank You!</TITLE>
</HEAD>
```

```
<BODY>
<H1>Thank You!</H1>
```

```
Thank You for contributing to this server. You may now
<A HREF="/default.html"> Return to Our Home Page</A> or
<A HREF="<HTMLFILENAME>"> Review Your Article</A>
```

```
</BODY>
</HTML>
```

The response file can include any insertion commands (REPLACE, BULLET, etc.) allowed in the FDML file. Directives (CREATEDOC, MENUDOC, etc.), of course, cannot be included, because in most cases this wouldn't make any sense. If you want to include directives in your response page, you probably need to look into using the CHAIN directive to allow multiple FDML files to process the incoming data.

By allowing commands in the response file, you can give the user immediate feedback on the text that they entered. This will allow you to personalize the responses by echoing data back to the user, and can also be used as a way for users to preview their documents before they are permanently submitted.

Allowing users to retrieve a list of articles that have been recently added to the server is a key advantage of NetCloak Pro. In fact, you may create multiple lists of recently added articles, which is one of the easier functions to implement for the server administrator.

Using Recent Articles Lists

First, using the RECENTLIST directive, like this:

```
<RECENTLIST>LinkText</RECENTLIST>
```

will automatically add the user entry to the internal database of recently added articles. See the Supplemental Directives section in the Reference Guide for details on the RECENTLIST directive.

To access the database of recently entered articles, you request a special URL to be returned by NetCloak Pro, which contains the all-uppercase text “RECENT” and ends in the suffix “.fdml” (Recent article URLs will work with any suffix mapped to NetCloak’s NFORMS action for forms-processing.) Recent article links can be used on any page on your server.

Linking to “RECENT.fdml” will display for the user a list of the documents added since the last time that user accessed the list. What happens is:

- The user clicks on the link to “RECENT.fdml”, and NetCloak Pro will search the internal database for the user’s machine address.
- Once found, the time and date of their last access is retrieved and compared with the recent articles list.
- Documents having a later time stamp than the user’s last access will be displayed in the list.
- Finally, the user’s last access time is updated with the current time.

There are a couple of variations on the recent article list URL that provide some additional functionality. The first is “RECENT_minutes.fdml”. This version of the RECENT command allows you to specify a time frame for what is considered a recent article. Replace “minutes” with a number of minutes. For example:

USING RECENT ARTICLE LISTS

- “RECENT_60.fhtml” would show the user articles added in the past hour.
- “RECENT_1440.fhtml” would show the user articles that are less than a day (1440 minutes) old.

You might need to do some multiplication to get the time frame you want (a week would be $7 \times 24 \times 60$, or 10080 minutes).

Maintaining Multiple Recent Articles Lists

You may allow users to access one or more of up to 26 lists of recent articles by appending a single letter A through Z to the end the command. This allows you to maintain lists for several systems on your server independently. If no list is specified, NetCloak will use the A list.

In the FDML, add a space to the opening RECENTLIST tag followed by the letter designating the list you wish to include articles. It will look like this:

<RECENTLIST C>

In an HTML document, create a link to a particular recent articles list by adding an underscore followed by the letter to the RECENT URL, like this:

RECENT_C.fhtml

Note that the URL to access a recent list must contain an underscore character ('_') before the list and the minutes value. However, the syntax of the RECENTLIST directive in your FDML files does *not* use an underscore — it uses a space character instead.

Use of specific lists can be combined with a time frame. For example, to link to the default, second, and a specific time frame of a third list, the syntax would be as follows:

RECENT_A.fhtml (or just RECENT.fhtml)

RECENT_B.fhtml

RECENT_C_60.fhtml

Here is an example that should make use of the RECENT command clearer. A section like this might be found on your server's home page:

USING RECENT ARTICLE LISTS

Click on one of these links to view articles entered...

```
<UL>
<LI><A HREF="RECENT.fhtml">
since I last checked</A>
<LI><A HREF="RECENT_60.fhtml">
in the Last Hour</A>
<LI><A HREF="RECENT_480.fhtml">
in the Last 8 Hours</A>
<LI><A HREF="RECENT_1440.fhtml">
in the Last Day</A>
<LI><A HREF="RECENT_4320.fhtml">
in the Last 3 Days</A>
</UL>
```

Finally, remember that each list can contain up to 256 links. If more articles than that have been added in the time frame specified (or since a particular user's last access), only the most recent 256 will be shown.

What Are Mail Actions?

In addition to processing an FDML file in response to a form submission or a Web page hit, NetCloak Pro also allows you to process an FDML file whenever a message is received from any e-mail address. The collection of information needed to associate an e-mail address with an FDML file is called a “Mail Action”.

NetCloak Pro can check any e-mail account located on any POP3-compatible mail server on the Internet. It can check for mail as frequently or infrequently as you like.

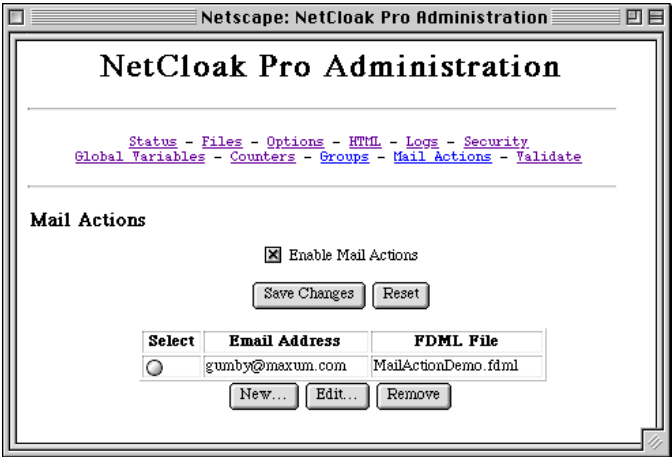


Figure 6: The Mail Action Web admin page

Using Mail Actions, you can have NetCloak Pro automatically check an e-mail address and create a new Web page, insert information into existing pages, store data, or even send another e-mail message, using the information contained in each new message. For instance, you can easily create Mail Actions to create online archives of your favorite mailing lists, automate routing of e-mail messages to the appropriate persons, or provide Web access to e-mail for members of your organization.

Setting Up Mail Actions

To create, edit, or remove Mail Actions, you use the “Mail Actions...” command in the NetCloak Pro CGI or access the “Mail Actions” link from the NetCloak Pro Web admin home page.

Because Mail Actions require NetCloak Pro to open TCP/IP connections to remote mail servers, a single control is provided to enable or disable all Mail Actions at once. To enable checking of all configured Mail Actions, check the “Enable Mail Actions” checkbox. In the CGI, checking is enabled as soon as you check the box. In the Web admin page, you must click the “Save Changes” button after checking the box. To disable Mail Actions, un-check the box (and, in the Web admin page, click “Save Changes”).

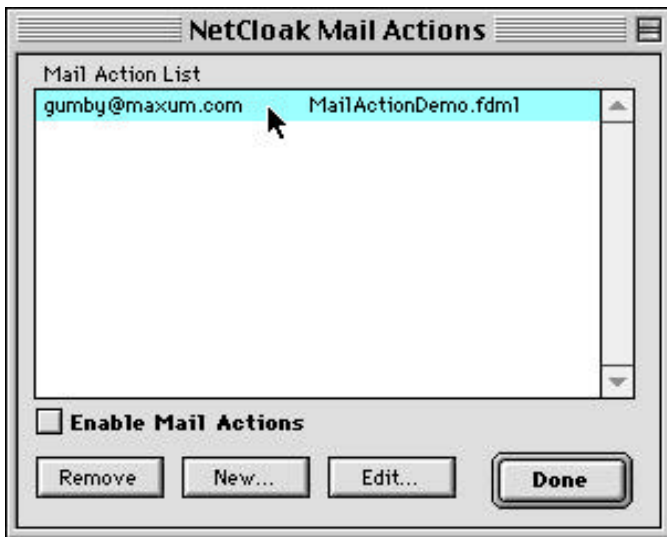


Figure 7: The Mail Action window

To edit an existing Mail Action, select it (in the CGI application window) or click the “Select” radio button for that action (in the Web admin page), then click the “Edit...” button. To create a new Mail Action, click the “New...” button. In either case, you are presented with the “Edit Mail Action” window (in the CGI) or Web admin page.

A Mail Action consists of three sets of information: which e-mail account to check, how often to check it, and what action to take when messages are received.

Email Account Settings

To specify which e-mail account you want NetCloak Pro to check, you must provide four pieces of information: the IP address or domain

MAIL ACTIONS

name of a POP3 mail server, the e-mail address of the account, the password needed to access that account, and whether or not NetClook Pro should use “APOP authentication”.

Netscape: NetClook Pro Administration

NetClook Pro Administration

[Status](#) - [Files](#) - [Options](#) - [HTML](#) - [Logs](#) - [Security](#)
[Global Variables](#) - [Counters](#) - [Groups](#) - [Mail Actions](#) - [Validate](#)

Edit Mail Action

Mail Server:

Email Address:

Email Password:

☐ Use APOP Authentication

Check Every:

FDML File:

Note: Mail action FDML files must be in the configured Root Folder:
Macintosh HD:Work:NetClook 3.1:NetClook 3.1 Package:

Figure 8: The Edit Mail Action Web admin page

Mail Server -- Enter the IP address (such as “198.64.78.12”) or domain name (such as “pop.maxum.com”) of a mail server in this field. The specified IP address or domain name must be that of a machine connected to the Internet (or is at least visible to NetClook Pro on your network), which is running a POP3 (“Post Office Protocol” version 3) mail server. That mail server must be the mail server that stores messages for the specified e-mail address.

Email Address – This field contains the fully-qualified address specification of the e-mail account from which messages will be retrieved. For instance, “chris@maxum.com”. This e-mail address must be recognized as a valid address by the configured mail server.

Email Password – Enter the corresponding password for the configured e-mail account into this field. In the CGI, the password characters will appear as clear text until you tab out of the field, when

they are converted to bullets. How the password field in the Web admin page is displayed depends on the particular browser you use.

Use APOP Authentication – With this box unchecked, the configured e-mail address and password are transmitted to the mail server as clear text, so that anyone “snooping” on the network will be able to read them. If the mail server supports it, using APOP authentication will encrypt the username and password during transmission, so that they cannot be easily discovered. Ask the mail server administrator if the configured mail server support APOP (or, just try it and see if it works).

The screenshot shows a dialog box titled "Edit Mail Action". It is divided into three sections: "Email Account", "Schedule", and "Action".
- The "Email Account" section has three text input fields: "Mail Server:", "Email Address:", and "Email Password:". Below these is a checkbox labeled "Use APOP Authentication".
- The "Schedule" section has a "Check Every:" text input field, followed by the word "Minutes" and a small up/down arrow spinner.
- The "Action" section has a button labeled "FDML File:".
At the bottom of the dialog are two buttons: "Cancel" and "OK".

Figure 9: The Edit Mail Action window

Schedule Settings

NetCloak Pro will attempt to log on to the mail server using the frequency you specify here. It can be configured to check for mail as often as once per minute, or only very infrequently. Select units of “Minutes”, “Hours” or “Days” from the popup menu (via the Web) or using the arrow buttons (in the CGI), and enter the number of units in the text field. For example, to check mail every 24 hours, enter “24” into the text field, and select “Hours”.

MAIL ACTIONS

You cannot select the time of day NetCloak will check mail. NetCloak Pro checks once per minute for any Mail Actions that need to run. So, if you create a Mail Action at 3:30 PM, and it is set to check mail every 24 hours, it will check that action's e-mail address at roughly 3:31 PM each day.

Action Setting

Finally, you must tell NetCloak Pro what to do when messages are received by the configured e-mail address. The way you do this is by specifying the path to an FDML file located in the configured Root Folder. In the CGI, click the "FDML File" button. You are presented with a standard Mac OS "open file" dialog box. Select the FDML file you want to use to process mail messages for the configured e-mail address. In the Web admin interface, you must enter the exact path to the FDML file, from the volume name down, in the "FDML File" text field. Enter the path using Mac OS file path conventions – use colons to separate folder names, and let the text field wrap the text automatically (don't insert return characters yourself).

Once you have entered all the information needed by the Mail Action, click "OK" (in the CGI) or "Save Changes" (in the Web admin page) to save the Mail Action. In the CGI, NetCloak Pro will display an error message if any fields are incorrectly filled out. No such error checking is performed in the Web admin interface, so be especially careful when editing Mail Actions this way.

Creating a Mail Action FDML File

All normal FDML directives and insertion commands are available for use in a Mail Action FDML file. Use CREATEDOC to create Web pages from mail messages; INSERTFILE to insert messages into existing pages; TEXTSTORE to maintain a tab-delimited text file of mail data; or SENDMAIL to generate messages to other e-mail addresses based on the received mail.

Where do the form field variables normally used in FDML insertion commands come from? NetCloak Pro automatically converts the header fields from each mail message into form variables that can be accessed using standard REPLACE commands. In addition, the body of each message is placed into a form field named "Body", which you can insert like this:

<REPLACE Body>

All mail messages will contain a few standard variables that you can depend on being present when using them in REPLACE commands. These are listed below.

From	The e-mail address of the sender of the message.
To	The e-mail addresses of the message's primary recipients; commas separate each address. The Mail Action's configured e-mail address will be in this list.
Subject	The "title" of the mail message.
Date	The date the message was originally transmitted, in the standard format required by the SMTP/POP3 protocols.
Body	The body of the message (not including headers), generated by NetCloak Pro.

In addition, most modern e-mail clients and servers will include the following optional variables in the message:

CC	The e-mail addresses of the message's secondary, "carbon-copy" recipients, separated by commas. This will only be present if secondary addresses exist.
Content-Type	Defines the format of the data contained in the e-mail message. Plain text messages have a content type of "text/plain", while HTML formatted messages have a content-type of "text/HTML".
Message-ID	A unique identifier for the mail message used by the mail server. This unique ID is usually in the form of an email address, like <0123abc1f332@maxum.com>, so it is guaranteed to uniquely distinguish this message from all other messages on all other mail servers.

For a complete list of all the possible mail message header fields, see the RFC documents describing the format of mail messages (RFC 822) and MIME headers (RFCs 1521 and 1522), available many places on the Internet.

Debugging Mail Actions

When you create a new mail action, it may not work the first time. NetCloak has special handling for mail action errors to help find problems specific to mail actions.

MAIL ACTIONS

If an error occurs when NetCloak tries to perform a mail action, it will create a file named MailAction.err in the NetCloak Files folder on the server. This file will have a description of the error, along with the name of the mail action and the date and time the error occurred.

Most errors specific to processing a mail action will result in the generation of a MailAction.err file. Some common problems include the specified mail account does not exist, having the wrong password or NetCloak being unable to locate the specified FDML file. Other FDML errors that would be displayed on a response page for a web-based system will also be recorded in the MailAction.err file.

When several errors occur, NetCloak will overwrite the MailAction.err file for each one. This means that there will only be one file and it will only contain information for the most recent error. If there are several problems (or several mail actions with a problem), you will have to troubleshoot and fix them one at a time.

Message Content Types

When NetCloak Pro receives a mail message in response to a Mail Action, it looks for a “Content-Type” field and a “Content-Transfer-Encoding” field in the header of the mail message. Messages with certain content types and transfer encodings are handled specially by NetCloak Pro.

If the content type of the message is not specified (no “Content-Type” field is present), NetCloak Pro assumes the message’s content type is text.

If the content type of the message is multipart/mixed or multipart/alternative, this means that the message contains either attachments or alternative ways of displaying the same message (most often, a plain text and an HTML version of the message). In this case, NetCloak will extract the text/HTML or text/plain part of the message, and then discard the rest. If both a text/HTML part and a text/plain part exist, NetCloak will use the HTML version of the message as the Body variable. Note that this means that NetCloak always discards attachments.

If a transfer encoding is not specified, 8-bit encoding is assumed. In other words, NetCloak does not modify the data in any way. If the

transfer encoding is either quoted-printable or base64, NetCloak properly decodes the message body.

Update Recent Database

The Update Recent Database menu command in the Configuration menu will scan all recent lists currently in memory, removing entries for files that no longer exist on the server. In other words, when you delete articles from your server, you can have them removed from the recent lists simply by selecting this menu item.

INSTALLATION

Installing NetCloak requires several steps, many of which are different depending on which web server you are using. The following sections provide an overview of the installation process, then specific instructions for several popular Mac OS web servers. If you are already comfortable with configuring your web server to use CGI or plug-in extensions you'll find NetCloak simple to install. Even if you are familiar with configuring actions, we suggest you skim this chapter for specific procedures and options unique to NetCloak.

If you have already downloaded a demo version and installed the NetCloak Introduction, you need only follow the steps for creating a suffix mapping and/or associated action and to validate the demo software. You will also need to validate the demo version by entering your serial number and validation code on the Validation page of the web administration.

An Overview Of Suffix Mappings And Actions

Each of the HTML pages you create using NetCloak commands must be processed by NetCloak, or "cloaked". Most web servers in use today make use of "suffix mappings" and "actions", which greatly simplifies using NetCloak (and many other server add-on tools) on your server.

An action is configured in the web server, and defines a method for handling a request. Most servers have built-in actions for "TEXT", which tells the server to return the requested file as text (ASCII) data, and "BINARY", which tells the server to send binary data. Servers also allow you to define your own actions which tell the server to hand off the request to a plug-in module or an external CGI instead of just sending the requested file.

A file suffix is the last few characters of the filename, and begins with a period. File suffixes are normally used to denote what type of information is contained in a file, such as HTML text or a GIF image. Common suffixes include: ".html", ".gif", and ".jpeg". A suffix mapping defines which action to be taken for a particular suffix. For example, you can configure your web server to use NetCloak as an action to process all files with the suffix ".html".

Exactly how you define suffix mappings and the action to be taken varies from server to server. Basic instructions for several different servers are included in the following sections. For detailed information

on configuring suffix mappings, see the documentation that came with your server.

No matter what server you are using, we suggest that you take the time to configure at least one suffix that uses NetCloak to process files. Remember that any page that you use NetCloak extension commands in will need to be named with the appropriate filename suffix. For example, if you define a suffix mapping of “.nclk” for NetCloak processing, any page that includes NetCloak commands must have a filename ending in “.nclk”.

Installing NetCloak

To install NetCloak, you must first place NetCloak and its associated files in the appropriate places on your server. Next, you must create a suffix mapping and an associated action to call NetCloak when pages are requested from your server.

Because NetCloak requires very little overhead, most Webmasters that use NetCloak simply make NetCloak the action responsible for processing pages with the suffix “.html”. Since all of your HTML pages most likely end in “.html” already, this will allow you to use NetCloak in any of your HTML pages without worrying about whether the page is cloaked or not.

Choosing the Plug-In or CGI Version of NetCloak

There are actually two versions of NetCloak included in the package: a Plug-In and a Common Gateway Interface (CGI) application. Most Mac OS web servers support WebStar API plug-ins, and best results are usually obtained with the plug-in. NetCloak imposes almost zero overhead when running as a Plug-In, ensuring peak performance from your web server.

Servers that support plug-ins include WebSTAR, WebTen, Quid Pro Quo, AppleShare IP and TeleFinder. Some notable servers that do not are Mac OS Personal Web Sharing, Microsoft Personal Web Server and NetPresenz. The NetCloak CGI is not as fast as the plug-in, but is compatible with web servers that do not support WebSTAR API Plug-Ins.

INSTALLATION

Installing NetCloak and Its Associated Files

The location of the NetCloak application or plug-in and the NetCloak support files is dependent on the web server application you are using. The NetCloak Plug-In file must go in the web server's plug-ins folder. This folder is normally named "Plug-Ins" and is usually located in the web server root folder. The location of the "NetCloak Files" folder used by the plug-in varies according to which web server you use — see the instructions below specific to your web server.

To install the CGI version of NetCloak, see the instructions in the Reference Guide.

Creating Actions and Suffix Mappings

Most web servers define actions and suffix mappings separately. WebSTAR, Quid Pro Quo and WebTen work this way, so you must first define an action which calls NetCloak, then assign that action to selected suffixes by defining suffix mappings (sometimes called "MIME mappings" or "MIME extensions"). The NetCloak plug-in will automatically define an action for itself named "CLOAK_PI" so you only need to worry about creating an appropriate suffix mapping in most cases.

Some web servers define actions and suffix mappings together. AppleShare IP defines MIME mappings which include both a file suffix and the method for handling requests for that file type. The Web Sharing control panel defines actions which are similar to AppleShare IP's MIME mappings.

For servers like WebSTAR and WebTen which support multiple virtual-hosted or multihomed Web sites served by a single server, you may need to define actions and suffix mappings for each site separately.

Accessing the Web Administration Interface

Configuration is most convenient through NetCloak's web-based administration pages. Both the plug-in and the CGI version can be configured this way, but the CGI can also be configured through the application's traditional user interface. Using the CGI to configure the plug-in is possible, but not recommended.

Once you have the plug-in and other NetCloak files installed and your web server running again, load the main administration page (the Status page) with the URL:

`http://your.server/pi_admin.nc1k`

You will, of course, need to replace “your.server” with the address of your web server. Normally there is an additional step—most web servers require a username and password to access any “pi_admin” URL. You may need to configure a username and password in your web server before you will be able to access the admin pages, or, for some servers, the server admin username and password will work.

Setting Default, Error and No Access Filenames

When a user requests a file which does not exist, or the user has insufficient privileges to access a file, web servers return some sort of error page. Similarly, when a user makes a request with a file path but no filename, web servers return a predefined “default” or “index” page.

When NetCloak handles such a request, it must perform the same function. If you are using the plug-in version of NetCloak with WebStar, WebTen or other servers that allow defining these files in the server, just make sure the “Ask Web Server” option is checked and NetCloak will automatically use the setting in the server. This will work with multi-homed or virtual host servers with different files or filenames for each site. If you have a server such as AppleShare IP or Web Sharing which do not use HTML files for these pages, you will need to create them and configure NetCloak with the filenames.

Validating NetCloak

When you purchased NetCloak, you received a serial number and validation code. If you installed a demo version of NetCloak you will need to enter these in order to remove the demo expiration date.

Load the Validation page of the NetCloak web administration. Enter your serial number and validation code in the spaces provided, then click the Validate button. If validation was successful, the page will have a message “Validation Complete”. It should look like figure xx. If the validation page is returned without the Validation Complete

INSTALLATION

message, be sure you are entering the serial number and validation code correctly.

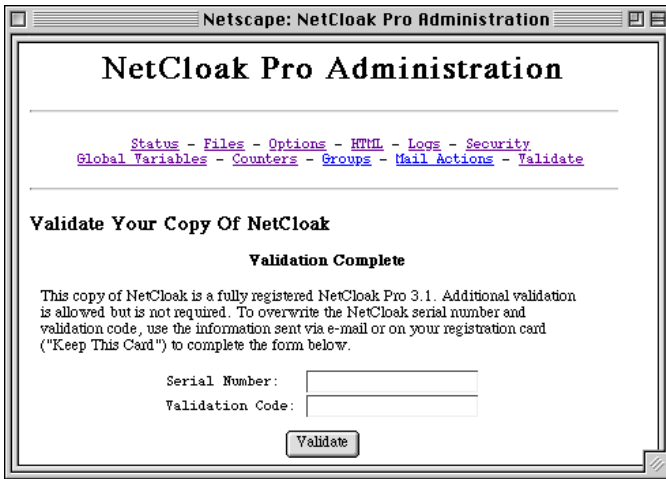


Figure 10: The Validate page of the NetCloak web admin.

Alternatively, you can use the Validate command in the File menu of the NetCloak CGI application. After validating the CGI, the Validate Plug-In command will be available. This can be used to validate the NetCloak plug-in, even if it is a demo version which has expired. Be sure to quit your web server before attempting to use the Validate Plug-In command.

Uninstalling NetCloak

Once you have created suffix mappings to use NetCloak, your server will always send files with that suffix through NetCloak. If you remove NetCloak, an error will result unless you reset the suffix mapping. If you are using WebSTAR, you can use the NetCloak CGI application's Auto-Map menu commands ".HTML files to BINARY Action", ".NCLK files to BINARY Action", and ".FDML files to BINARY Action" to easily reset these suffix mappings.

If you are using a web server other than WebSTAR, you will need to manually remove the suffix mapping. See your web server documentation for instructions on editing suffix mappings and default suffix mappings. If you manually created a suffix mapping for a suffix

other than “.html”, “.ncl”, or “.fdml”, then you will need to manually reset or delete it.

Installing NetCloak with WebSTAR

The “NetCloak Files” folder is normally located in the WebSTAR folder, which is also ordinarily the Root Folder of your web site.

1. Copy the “NetCloak Plug-In” file from the “NetCloak 3.1” folder to the “Plug-Ins” folder in the same folder as the WebSTAR application.
2. Launch WebSTAR.

If you want to process all HTML files through NetCloak, also follow these steps:

3. Launch the NetCloak CGI application.
4. Select “.HTML Files to Plug-In” from the “Auto-Map” menu.
5. Quit NetCloak.

You can also change the suffix mapping for “.html” to the action “CLOAK_PI” manually with WebStar Admin instead of using the NetCloak CGI application.

The NetCloak Pro Plug-In automatically registers the “NFORMS_PI” action with WebSTAR and maps the “.fdml” suffix to that action. No additional steps are necessary, unless you want to use other filename suffixes with your FDML files.

Installing NetCloak with AppleShare IP

These instructions apply to **AppleShare IP 5.x**.

With AppleShare IP, the “NetCloak Files” folder used by the NetCloak Plug-In is located in the “Extensions” folder inside the System Folder of your server.

1. Copy the “NetCloak Plug-In” file from the NetCloak 3.1 folder into the “Plug-Ins” folder inside the Web Folder on your AppleShare IP server.
2. Launch AppleShare IP Admin.
3. Open the “Web & FTP MIME Types...” window by selecting it in the Server menu.
4. Click on the line with the suffix “.html”, then click the Edit button.

INSTALLATION

5. Click on the pop-up menu for Action: and select Execute...
6. After the file open dialog box appears, select the "Plug-Ins" folder in the list and click the open button, then select NetCloak Plug-In in the file list and click the Open button again to select the NetCloak plug-in.
7. Click the OK button in the Edit MIME Type dialog box, then click Save. Click the OK button in the alert that appears after you click the Save button.
8. Open the Web & File Server Settings window by selecting it in the Server menu.
9. Select the Web Plug-Ins pane by using the pop-up menu at the top of the window.
10. Check the Enable Plug-Ins checkbox.
11. Change the Maximum Plug-In Memory Usage setting to at least 2048 KB.
12. Click the Save button to save the changes, then close the window
13. Stop the AppleShare IP Web & File Server, then re-start it.

You must manually create a MIME mapping for forms processing if you have the Pro version of NetCloak.

1. Launch AppleShare IP Admin.
2. Open the Web & FTP MIME Types... window by selecting it in the Server menu.
3. Click on the line with the suffix ".html", then click the Duplicate button.
4. Change the suffix to ".fdml".
5. Click on the pop-up menu for Action: and select Execute...
6. After the file open dialog box appears, select the "Plug-Ins" folder in the list and click the open button, then select "NetCloak Plug-In" in the file list and click the Open button again to select the NetCloak plug-in.
7. Click the OK button in the Edit MIME Type dialog box, then click Save. Click the OK button in the alert that appears after you click the Save button.
8. Stop the AppleShare IP Web & File Server, then re-start it.

These instructions apply to **AppleShare IP 6.1 or later**.

With AppleShare IP, the “NetCloak Files” folder used by the NetCloak Plug-In is located in the “Extensions” folder inside the System Folder of yourserver.

1. Copy the “NetCloak Plug-In” file from the NetCloak 3.1 folder into the “Plug-Ins” folder inside the Web Folder on your AppleShare IP server.
2. Launch MacOS Server Admin.
3. Click on the web server icon, and select Configure Web Server from the popup menu.
4. Click the MIME Types tab in the Web Server Settings window.
5. Click on the line with the suffix “.html”, then click the Edit button.
6. Click on the pop-up menu for Action: and select Handle File Using Plug-In...
7. Select the NetCloak Plug-In from the list, then click the Select button.
8. Click the OK button in the Edit MIME Information dialog box.
9. Click the Plug-Ins tab of the Web Server Settings window.
10. Check the Enable Plug-Ins checkbox.
11. Change the Maximum Plug-In Memory Usage setting to at least 2048 KB.
12. Click the Save button to save the changes, then close the Web Server Settings window.
13. Stop the AppleShare IP Web & File Server, then re-start it.

You must manually create a MIME mapping for forms processing if you have the Pro version of NetCloak.

1. Launch MacOS Server Admin.
2. Click on the web server icon, and select Configure Web Server from the popup menu.
3. Click the MIME Types tab in the Web Server Settings window.
4. Click on the line with the suffix “.html”, then click the Duplicate button.
5. Change the suffix to “.fdml”.
6. Click the OK button in the Edit MIME Information dialog box.
7. Click the Save button to save the changes, then close the Web Server Settings window.

INSTALLATION

8. Stop the AppleShare IP Web & File Server, then re-start it.

Installing NetCloak with WebTen

The location of the NetCloak Files folder used by the NetCloak Plug-In depends on which version of WebTen you are using. In all cases, the folder is in the same location as the actual WebTen application:

<i>Version</i>	<i>Location of the NetCloak Files folder</i>
1.1.1	Main WebTen folder
2.0 – 2.1.3	WebSites folder
2.1.4 – 2.1.x	Tenon Libraries:Tenon Kernel Libraries folder
3.0 +	WebSites folder

1. Copy the NetCloak Plug-In file from the NetCloak 3.1 folder to the Plug-Ins folder in the WebTen folder.
2. Launch WebTen.
3. Launch your web browser if it is not already running.
4. Access your server's Administration Serve" home page (at http://www.yourserver.com/webten_admin) with your browser.

For WebTen 2.1 and earlier only:

5. Click "Advanced".
6. Set the "MyopicPlugInMode" setting to "Off".
7. Click "Save Advanced Settings".

If you want to process all HTML files through NetCloak, also follow these steps:

8. Click the Action Handlers button in the Administration Server home page.
9. Type "CLOAK_PI" in an empty Action field.
10. Type "NETCLOAK-PLUGIN" in the corresponding Action Handler field.
11. Click the Save Handlers button.
12. Click the MIME Extensions button.
13. Click the Built-In Extensions button.
14. Select CLOAK_PI from the Action popup menu for the .html extension.

15. Click the Save MIME Extensions button.

Create a MIME extension for forms processing if you have NetCloak Pro:

1. Click the Action Handlers button in the Administration Server home page.
2. Type "NFORMS_PI" in an empty Action field.
3. Type "NETCLOAK-PLUGIN" in the corresponding Action Handler field.
4. Click the Save Handlers button.
5. Click the MIME Extensions button.
6. Type ".fdml" in an empty Extension field.
7. Select "NFORMS_PI" from the Action popup menu for the .fdml extension.
8. Click the Save MIME Extensions button.

Installing NetCloak with Other Web Servers

If you use web server software that is not listed, please consult your web server's documentation for information on configuring it for use with NetCloak. In particular, check for support for WebSTAR API plug-ins, and look for information about setting up file "suffix mappings" and creating "actions" or "action handlers".

The NetCloak plug-in automatically registers an action called "CLOAK_PI", and registers a suffix mapping which maps the ".nclk" file suffix to the "CLOAK_PI" action. You should also be able to modify the suffix mapping for ".html" files so that they are handled by the "CLOAK_PI" action (and thus the NetCloak plug-in).

The NetCloak CGI must be configured manually. You should create an action named "CLOAK" that defines a path to "NetCloak.acgi" as its handler, and then create a suffix mapping that maps the suffix ".nclk" (and optionally ".html") to the NetCloak action.

The NetCloak Pro plug-in additionally registers an action called "NFORMS_PI" and a suffix mapping which maps the ".fdml" suffix to the "NFORMS_PI" action. If you wish to use fdml template files that have a suffix other than ".fdml", you will need to manually create a corresponding suffix mapping.

INSTALLATION

The NetCloak CGI must be configured manually. For instructions on installing the NetCloak CGI with popular web servers, see the Reference guide. You should create two actions, named “CLOAK” and “NFORMS” that both define a path to “NetCloak.acgi” as their handler, and then create suffix mappings that map the suffix “.ncl” (and optionally “.html”) to the “CLOAK” action, and the suffix “.fdml” to the “NFORMS” action.

Installing NetCloak Pro with Personal Web Sharing

Mac OS Personal Web Sharing does not currently support WebSTAR API plug-ins, so you will only be able to use the CGI version of NetCloak. See the section “Using the NetCloak CGI” for instructions on installing the CGI version of NetCloak.

When you purchased NetCloak you received a serial number and validation code that make it easy to upgrade to future versions. To upgrade NetCloak, download the current demo version from the Maxum Web site, install it, then access the Validate page of NetCloak's web administration interface. Enter the serial number and validation code from your registration card stub or e-mail receipt and click Validate. A message indicating that validation is complete will then appear on the page indicating that the expiration of the demo has been removed.

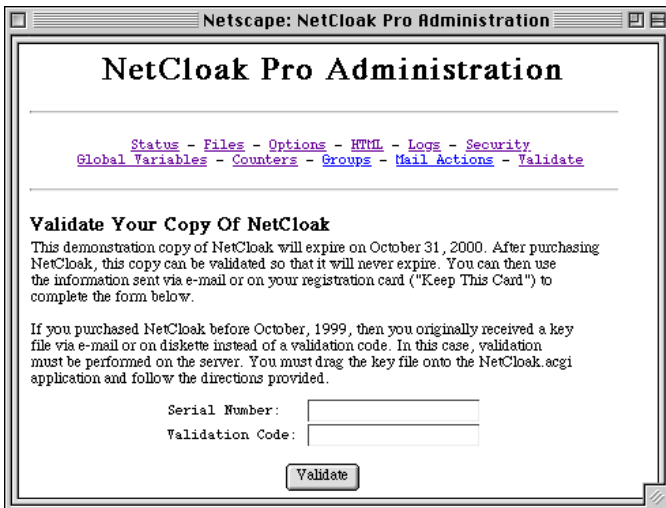


Figure 24: The Validate page of the NetCloak web admin.

If you are running the CGI version of NetCloak, you may also use the Validate command from the File menu.

At some point, at least one year from purchasing NetCloak, your serial number and validation code will no longer work with new versions of NetCloak. Software that you have successfully validated will continue to run, but new versions will require that you purchase an upgrade from Maxum. Upgrading and ordering information will be available on our web site along with the new software.

If you purchased a version of NetCloak before 3.1, you may have received a key file on diskette or via e-mail instead of a serial number and validation code. To validate new software with the key file, drag and drop it onto the NetCloak CGI application in the Finder.

UPGRADING NETCLOAK

Once validated, the NetCloak CGI can validate the plug-in using the Validate Plug-In command in the File menu. When you choose this menu item, you will be asked to locate the NetCloak plug-in to be validated using the standard file dialog. Note that you will have to quit your web server application before validating.

NetCloak has a number of options and parameters that you can configure that control how it serves, creates and logs files on your server. NetCloak provides a convenient web-based interface for setting these as well as editing counters, variables and all of the other administrative information. The CGI application also has a traditional GUI which provides the same functions.

The web-based administration pages are the recommended method of configuring the NetCloak plug-in. You may also use the web admin pages with the NetCloak CGI, but you must first enable access using the “Enable web administration” checkbox in the Security tab of the Configuration window (see below).

To access the browser-based web admin pages, access the following URL in your browser (replacing “your.domain.com” with your server’s actual domain name or IP address):

`http://your.domain.com/pi_admin.ncl`

Unless you have already entered a valid username and password for the pi_admin realm on your web server, your browser will display a dialog box requesting a username and password. Most Macintosh web servers, including WebStar and WebTen, automatically require a username and password to access any URL containing “pi_admin”, so that only authorized users may change plug-in configuration settings. If you have not previously logged in as an admin user, and you aren’t presented with a login dialog when accessing this URL, then your server does not automatically require a password for pi_admin URLs. You should immediately create one before continuing. Consult your web server’s documentation on how to create a realm. Failing to protect pi_admin URLs represents a severe security risk for your web server.

Once you have logged in as an admin user, you are shown the main NetCloak status page. Links at the top of the page allow you to navigate to other admin pages.

Note for AppleShare IP users: AppleShare IP 5.0 does not handle special URLs which do not represent actual files in the web server root folder, such as the NetCloak admin URL. If you access the above URL, you will be presented with a “file not found” error message. To work around this problem, you can create an empty file named “pi_admin.ncl” in your Web root folder. The file must be *completely*

CONFIGURING NETCLOAK

empty (zero bytes in length). With this file present, AppleShare IP will ask NetCloak to process it. NetCloak will then ignore the file and serve its internally-generated admin web page instead.

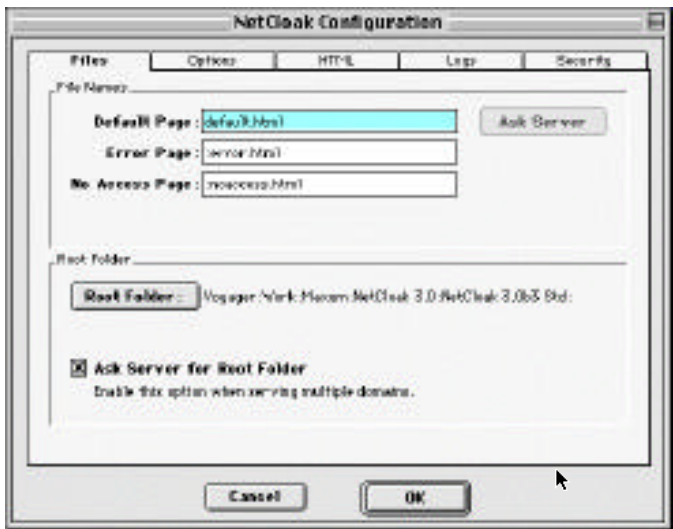


Figure 11: The Files tab of the NetCloak Configuration window.

When using the CGI version of NetCloak, you may alternatively use the commands in the “Configuration” menu in the NetCloak CGI application. The “Configure...” command displays the NetCloak Configuration window. While it is possible to use the CGI’s Configuration window to administer the plug-in, this is not recommended.

Files

The first tab of the Configuration window allows you to set the root folder and special file names needed by NetCloak.

File Names

There are three special files NetCloak requires to handle certain HTML requests.

Default Page: The name of the page NetCloak will look for when the user specifies only a directory (including the root domain). For example, if the default page is set to “HomePage.html”, then NetCloak

will look for that filename by default when only a directory is specified by an incoming URL request.

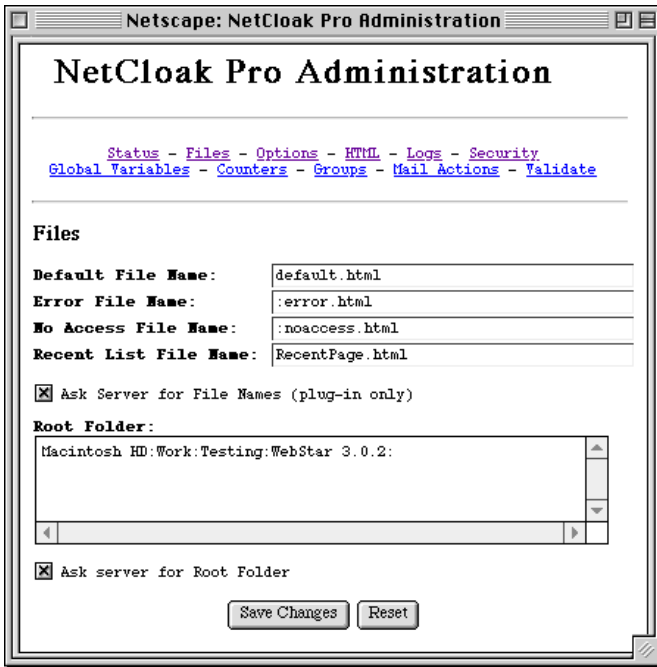


Figure 12: The Files page of the NetCloak Web admin interface.

Note that this field should *not* begin with a colon (':') character, since this field specifies a filename, not the path to an actual file. The filename in this field is appended to the incoming URL.

Error Page: When NetCloak is unable to find a file that matches an incoming URL, the Error Page is returned. This field specifies an actual file to be returned; thus, it contains the path to the file, relative to the configured Root Folder. If the error file is in the Root Folder, then this field should begin with a colon (':') followed by the name of the file.

If the field does not begin with a colon, then NetCloak looks for the error file in the same folder as the requested file. Although this is an unusual setup, it can be used to provide a custom error page for every folder on your web site.

A CGI application can also be specified as the error file, in which case NetCloak will call the CGI to handle the bad request. If the error file is

CONFIGURING NETCLOAK

suffix-mapped to a “back-end” CGI or plug-in in the “NetCloak.bsm” file, NetCloak will also call the back-end CGI or plug-in to process the error file. In either case, any NetCloak commands in the text returned from the “error CGI” are processed before returning the response to the browser.

When serving multiple virtual domains from your web server, the named Error Page must exist in each virtual domain’s root folder. Since NetCloak can resolve aliases, you may simply create an alias to an Error Page in each virtual domain’s root folder.

No Access Page: The “No Access” page is returned when a user is denied access to a page because their IP address is denied or because they fail to enter a correct username and password for a password protected page. This field, too, represents the path to an actual file, so it should also begin with a colon, followed by the filename, if it represents a file in the web server Root Folder. As with the Error Page setting, this field may contain the path to a CGI, or the path to a file which is suffix-mapped to a back-end CGI or plug-in.

When serving multiple virtual domains from your web server, the named No Access Page must exist in each virtual domain’s root folder. Since NetCloak can resolve aliases, you may simply create an alias to a No Access Page in each virtual domain’s root folder.

Ask Server – (CGI only) Most web servers also have their own settings for the three special file names. In particular, WebSTAR and WebTen provide access to these filenames via AppleScript, so NetCloak can retrieve the file names directly from either of these two web servers when you click this button. The button is only enabled if one of these two servers is up and running on the same machine along with NetCloak. If you are using another web server, this button will not work, and you will have to set the filenames manually.

Ask Server for File Names – (Plug-In only) Some web servers allow the administrator to configure separate default, error, and no-access files for multiple virtual domains. The NetCloak plug-in can get the names of the three files directly from the web server, on the fly, as each request is processed. This lets it use the correct file names for each request to a virtual domain. Check this checkbox in the NetCloak web admin “Files” page to configure NetCloak this way. If you want the NetCloak plug-in to always use the file names configured in NetCloak, uncheck this checkbox.

Root Folder

Choose the folder NetCloak should serve files from. To select the folder, click on the “Root Folder” button and choose the folder with the standard file dialog box that appears.

The selected Root Folder is used as the root folder for all NetCloak commands which accept a relative path, including macros, back-end CGIs and NetCloak aliases.

In most cases, the Root Folder will be the same folder as your web server’s root folder. The ability to select the Root Folder has two important advantages. First, when using the NetCloak CGI, this allows you to put the CGI application anywhere you wish, for example, in a “cgi-bin” folder, while serving from the web server root. In addition, the NetCloak root folder is not required to be the same as the server root folder. For example, if you are concerned about security, you can put your NetCloak files into a separate folder hierarchy from the main server folder structure. This will ensure that files that are supposed to be cloaked are never inadvertently served by your server without NetCloak processing them. Since the cloaked files will be outside of your web server folder, the server will never be allowed to serve them.

Important Note: This assumes you are running a server that restricts access to a web server root folder and that you have not placed any Finder aliases in your server root folder that would permit access to the NetCloak -defined root folder. It also does not take into consideration other CGIs that might be able to serve files outside the web server root.

Ask Server For Root Folder -- If the server supplies a root folder as part of the CGI AppleEvent or WebSTAR API request, and this option is checked, NetCloak will disregard the configured Root Folder setting and serve from the folder specified by the web server. Many web servers support this parameter, and selecting this option is the best way to ensure seamless file services when using NetCloak in conjunction with your server. This is especially true of servers that support multiple root folders for different domains. In this case, the folder to serve from may be different for each request that is processed by NetCloak. When this option is checked, NetCloak will always serve the file from the appropriate folder, as intended.

CONFIGURING NETCLOAK

Note that when this option is checked, NetCloak will use the root folder specified by the server for each request that it receives. NetCloak will still use the folder selected with the “Root Folder” button to locate NetCloak aliases and other files that are read during startup or outside of any HTTP connection request.

Options

Options

Server Options:

Server Name:

Last Modified: ☒ Exclude from header
☐ Send file's last modified date
☐ Send current date/time

☐ Include "Pragma: no-cache" for dynamic pages

Character Translation:

When Serving: ☒ None ☐ MacRoman to ISO-Latin-1 ☐ Convert to HTML Entities	When Saving: ☒ None ☐ ISO-Latin-1 to MacRoman ☐ Convert from HTML Entities
--	---

When Creating Files:

Creator Signature:

When Saving Duplicate Files:

☒ Smart Resubmit
☐ Always create a unique filename
☐ Always overwrite old file

Figure 13: The Options page of the NetCloak web admin

NetCloak supports several features that allow you to improve how your server behaves in different situations and for different browsers. To select these preferences, use the HTTP/HTML tab of the configuration window.

Server Options

Server Name -- The “Server Name” field is used by NetCloak to build the HTTP header returned to the client, and tells the client what server software is being used. The server name can be important for several reasons, and you should set this field carefully.

First, this field is used by many Web crawlers (also known as “robots”) for studies that determine what software is being used by whom on the Internet. By correctly returning the name of your server, you are helping to promote the Mac OS platform and the server software you have chosen. This is not a trivial issue, and we strongly urge you to enter the correct name of your server software and the name “NetCloak”. If you choose, you may list the names of other server applications, and they should be listed in order of importance, with the server itself always listed first.

It is not necessary to include the version numbers of the server, NetCloak, or any other server application. In fact, for security reasons, it is a good idea to leave the version number off. However, if you like, you may list the version numbers. If you choose to do this, the version number should follow the name of the application, with a slash (“/”) between them, and no spaces. For example, you could set this field to: `webSTAR/3.0.2 NetCloak/3.1`

Complete information on the server field can be found in the HTTP/1.1 specification, RFC 2068, section 14.39.

Last Modified -- Another field of the HTTP header is the “Last-Modified” date of the file. This field presents special issues when using NetCloak, because NetCloak is constantly changing the contents of files. However, it is important to use this field so that proxy servers and Web browser will correctly cache cloaked pages, and so that search engines will properly update their indexes when you update the contents of a page.

To solve this problem, there are three options:

Exclude From Header: When you select this option, NetCloak will not include the Last-Modified field in the header at all. This is a legal option, and is provided for in the HTTP specification. By choosing this, you are essentially leaving caching decisions up to proxies and browsers that access the page. It is not

CONFIGURING NETCLOAK

recommended, though, because caching is much more effective when the last-modified date is present. In addition, search engines often will not properly index pages that have no last-modified date.

Send File's Modified Date: If your site does not make excessive use of NetCloak's dynamic commands, your best option is to return the actual date the last time the file was modified. This is the same thing your server would do if the file was being served without NetCloak. This option, along with "Pragma: No-Cache for dynamic pages" is the preferred setting. Your pages will be crawled and updated correctly by search engines, but proxy servers and Web browsers will not cache them.

Send Current Date/Time: For pages that are very dynamic, and that should always be loaded from the server, use the "Send Current Date/Time" option. This will always send the date and time that the page is being served as the files last modified date. This may seem like a trick, but it is the most reasonable setting, in many cases. NetCloak modifies files as they are being served, so reporting the current date and time is, in fact, accurate.

Note that this default setting can be overridden on a page by page basis using the SET_NETCLOAK configuration command within your pages. You might want to do this if most of your pages are relatively static, with only a couple of very dynamic pages used on your site. In this case, you could set the default to "Send File's Modified Date" but use the SET_NETCLOAK command to send the current date and time for the very dynamic pages.

"Pragma: No-cache" for dynamic pages — Caching proxy servers and most Web browsers sometimes will cache a page even when the Last-Modified setting is set to "Current Date/Time". Checking this checkbox causes NetCloak to include a "Pragma: No-cache" field in the HTTP response header when the page contains any NetCloak command that causes the page's content to change with every request. This includes all SHOW, HIDE and INSERT commands, and many others. When "Pragma: No-cache" is present in the response header, HTTP 1.0 and 1.1 compliant proxy servers and browsers will not cache the page, regardless of the Last-Modified date or any other HTTP header field.

Character Translation

When Serving -- NetCloak will convert Mac Roman extended characters to ISO-Latin-1 character encoding or HTML entities. For US-ASCII only Web sites, leave this option on “No Translation”, but if you use extended characters, you can select an appropriate translation and edit your documents using the standard Mac OS character set. When these pages are served, the characters will be converted as specified so that they appear correctly in the user’s browser, regardless of the platform or operating system they are using.

When Saving -- NetCloak will convert ISO-Latin-1 character encodings or HTML entities in submitted form data to Mac Roman extended characters in created or updated files according to the selected option:

None — Performs no character conversion.

ISO-Latin-1 to MacRoman -- Convert “extended” ASCII characters (with ASCII values greater than 128) from the ISO-8859-1 (ISO-Latin-1) character set to the MacRoman character set, so that the file is in a standard Macintosh text format. When such characters are viewed in a Macintosh text application, they will appear as the expected Mac extended characters (the apple symbol, the bullet symbol, etc.). When viewed in a web browser, however, unexpected “garbage characters” may appear—particularly if the browser is running on something other than the Mac OS.

Convert from HTML Entities -- When this option is selected, extended characters in the form data are converted into the HTML-defined “entity” codes (such as “ ”), so that they appear correctly in HTML documents viewed in any Web browser.

For all of these options, the only way to know for sure what will work best for you is to test your site with a variety of browsers. There is (unfortunately) a wide variety of ways in which browsers treat extended characters, as well as the caching of pages based on the last-modified date. Decide what browsers are important to support at your site, download them, and try the various options to determine the best selection.

CONFIGURING NETCLOAK

When Creating Files

NetCloak sometimes needs to create text files on your server. Most often, it is creating log files or files that contain configuration information such as the list of defined groups. Files will be most commonly created when processing forms with NetCloak Pro. When it needs to create a text file, you can specify what application opens the file when it is double-clicked.

Creator Signature – Enter into this field the four-character creator code of the application that you wish to use to edit text files. By default, NetCloak uses the signature 'R*ch', the creator signature of BBEdit. If you don't know the creator signature of your favorite text-editing application, use the "Same Creator As..." button.

Same Creator As... -- Clicking this button presents a standard file dialog box, which lets you select any text file. NetCloak will then enter the creator signature of the selected file into the "Creator Signature" field automatically.

HTML

AutoMacros

Very often, a web server administrator wants to include a common header or footer on every page on their web site. NetCloak makes this a snap with its MACRO and MODEMACRO commands. However, this requires you to insert the macro command into the top and bottom of every page by hand. If you have hundreds or thousands of existing pages, this becomes an daunting task. Also, you cannot add macro commands to pages whose content you do not control, including pages that are generated on-the-fly by back-end CGI's or plug-ins.

In those situations, you can use NetCloak AutoMacros. As the name suggests, AutoMacros are macros that are automatically added to Web pages processed by NetCloak. You may define a header AutoMacro and a footer AutoMacro.

Header – Enter into this field the name of a macro contained in the "NetCloak.macros" file, or the path to a macro file, using the exact same syntax as you would for the filepath parameter in a MACRO or MODEMACRO command. When this field is filled in, NetCloak inserts an equivalent MODEMACRO command into each cloaked page, just

after the `<BODY>` tag that begins the page content. If no `<BODY>` tag is present in the page, NetCloak looks for a `</HEAD>` tag, and then a `<HTML>` tag. If none of these tags is present, NetCloak inserts the header AutoMacro at the top of the page.

Footer – Similarly, this field contains the name or filepath of a MODEMACRO to be inserted into each cloaked page, just before the `</BODY>` tag that ends the page content. If no `</BODY>` tag is present, it looks for a `</HTML>` tag. If neither tag is present, the footer AutoMacro is added to the bottom of the page.

If you do not wish to use AutoMacros, then leave these two fields empty.

Default Date/Time Format

Date – Use this popup menu to set the default date format used by `INSERT_DATE` and other NetCloak commands that insert dates. The first item in the menu displays your server's current Short Date format, and the second item displays your server's current Long Date format. The Short Date and Long Date formats are defined in the Mac OS Date & Time control panel on the server. See the Date and Time control panel's "Date Formats" setting to configure the short and long date formats. The default date format may be overridden within each NetCloak date command that has a format parameter.

Time – This popup menu displays the Short Time format and the Long Time format currently configured on your server. When the Short Time menu item is selected, all times inserted by NetCloak commands such as `INSERT_TIME` will be shown as hours and minutes, without seconds. When the Long Time format is selected, times will be displayed with seconds. The time formats displayed in the popup menu are configured in the Mac OS Date & Time control panel on the server. The default time format may be overridden within each NetCloak time command that has a format parameter.

Caching

Enable Dynamic Caching NetCloak supports a dynamic file cache that will automatically store your most frequently accessed files in RAM for faster retrieval and serving. The files NetCloak will store in RAM are selected based upon the amount of available RAM and the frequency

CONFIGURING NETCLOAK

with which files are accessed. To disable this feature, uncheck this box.

Flush Dynamic Cache — Use this button (CGI) or link (in the Web admin interface) to immediately force NetCloak to remove all cached files from its dynamic cache. This forces NetCloak to re-read all files from disk again, and frees up RAM in the application heap.

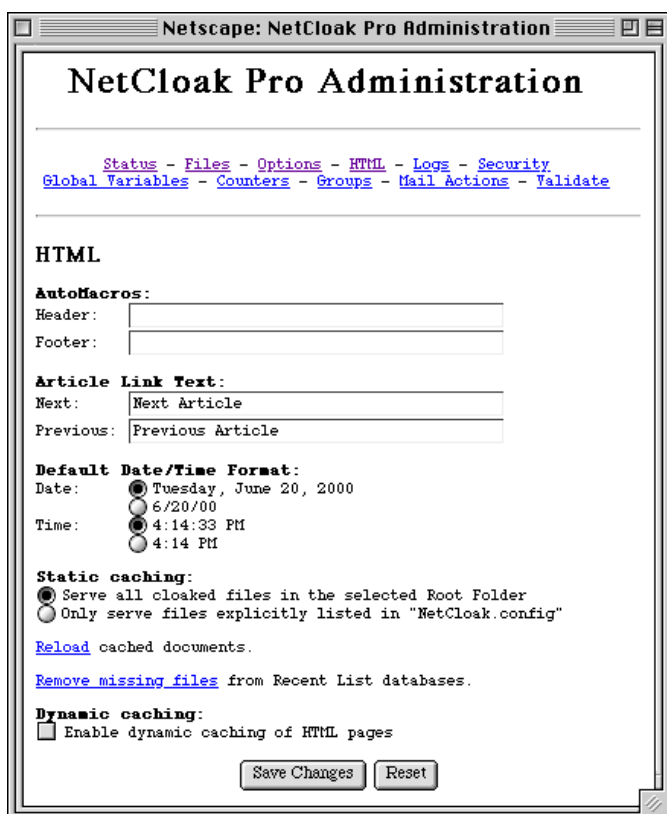


Figure 14: The HTML page of the NetCloak Web admin interface.

Reload Documents — Use this button (CGI) or link (in the Web admin interface) to tell NetCloak to re-read all files that are normally read and loaded into RAM at startup. This includes:

- all files listed in NetCloak.config
- NetCloak.groups
- NetCloak.macros

Logs

NetCloak can maintain a log in addition to the one created by your web server. Information included in the basic log is date and time of the request, status, the user's IP address or domain, the file requested, and the size of the response. The NetCloak extended log adds the username, password, referrer, and browser type. This extra information is added at the end of each line, so it should not interfere with log analyzing programs that expect only basic information.

One advantage to using the NetCloak log is that it will log only your HTML pages. When a client requests a page with multiple images on it, your web server log will contain several entries, one for the HTML page and another for each of the images linked into the page. This redundancy is often unneeded, and causes your log files to quickly become very large. In addition, the web server incurs the overhead associated with logging each page served multiple times.

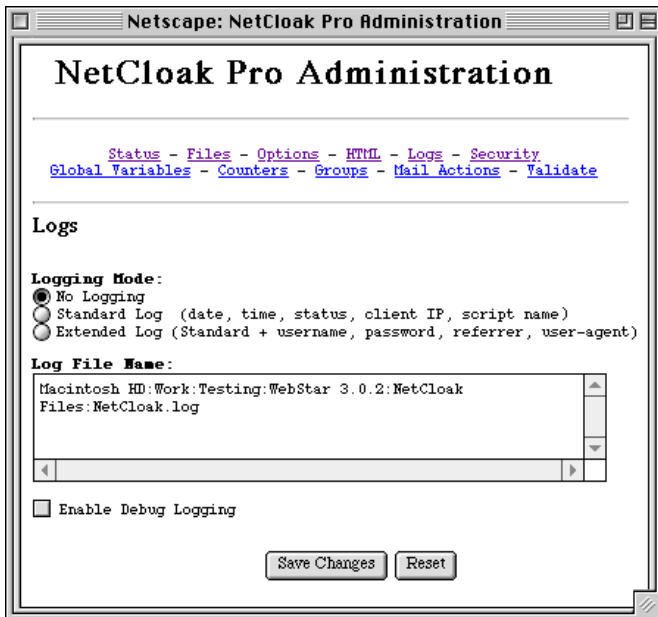


Figure 15: The Logs page of the NetCloak Web admin interface.

By shutting off your web server log, server performance will improve and your logs will be more manageable. The NetCloak log can then be used to show you who has visited your site and when. Of course,

CONFIGURING NETCLOAK

NetCloak will only log those file requests that it processes, so be aware that some site access will not be logged at all. This may not be appropriate for all Web sites, but for those whose logging requirements aren't as strict it is an excellent way to improve performance and simplify site management.

To set logging preferences, choose the "Logs" tab of the Configuration window.

You can choose whether to maintain a standard log, an extended log, or skip logging entirely. You can also choose to store the log information anywhere on your hard drive by clicking the "Log File" button and selecting a file to keep the log.

When strange problems occur, you can "Enable Debug Logging" in NetCloak. This causes NetCloak to create a file named "NetCloak Debug Log" located in the NetCloak Files folder, which contains very detailed debugging information about NetCloak processing. This information is mostly useful to engineers here at Maxum when debugging software problems, so you should ordinarily leave this option disabled unless instructed to enable it by Maxum tech support personnel.

Security

NetCloak provides several security options designed to make your server as secure as you need it to be while giving you flexibility that is not possible with your web server alone. Much of NetCloak's security benefits are found in the commands themselves, but there are also several options you can select from on the "Security" tab of the Configuration window.

General

Don't serve files with creator code 'WWWΩ' – This option lets you specify that Mac OS files created by certain applications are not allowed to be served through NetCloak. The default setting is to prevent any file created by WebSTAR (with creator code 'WWWΩ') from being served, which protects the sensitive information contained in the WebSTAR Settings and WebSTAR.pass files created by WebSTAR. If you use some other web server software, you may want to change this setting to the creator code of that application. Utilities such as

ResEdit or shareware such as Snitch or FileTyper allow you to view and/or modify the creator codes of files.

Enable web administration – (CGI only) By default, the NetCloak CGI does not allow access to the Web browser administration pages. To be able to edit configuration settings of the NetCloak CGI from any Web browser, check this checkbox. Web administration is always enabled in the NetCloak plug-in, since it is the only way to access the plug-in's configuration settings.

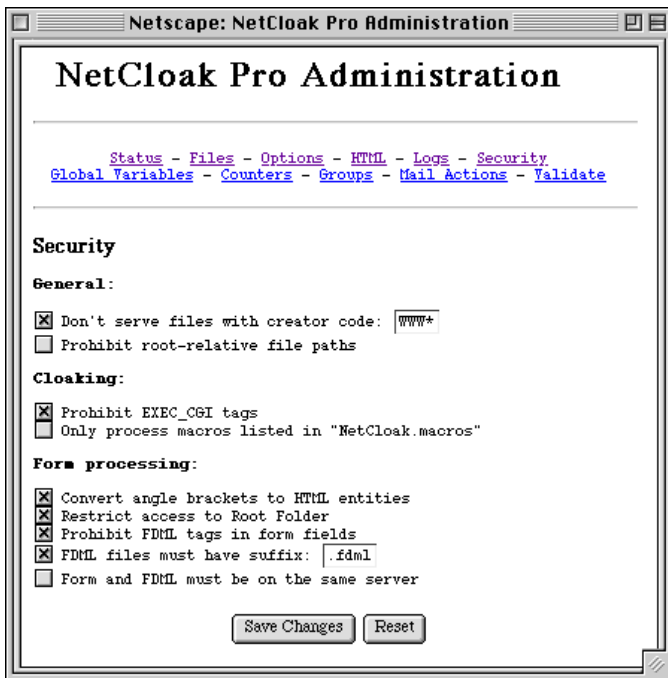


Figure 16: The Security page of the NetCloak Web admin interface.

Prohibit root-relative file paths in tags – This security setting is most useful to web server administrators who host Web pages for users who are allowed to modify their own pages. When this box is checked, file paths in all NetCloak commands are restricted to files in the same folder as the current page, or lower, using “file-relative” paths. In other words, only paths that do not begin with a colon or slash are allowed. This prevents any cloaked page from accessing information stored in any page outside the same folder. This setting is

CONFIGURING NETCLOAK

useful to few users and should ordinarily be left unchecked, which is the default setting.

Cloaking Pages

Only serve files listed in “NetCloak.config” -- When this option is checked, NetCloak will not serve a requested file unless it is aliased by a line in the “NetCloak.config” file. This represents the ultimate level of security provided by NetCloak. In this mode of operation, users visiting your web site can only see aliased files, and the aliases used in your URLs can give the illusion of an entire folder hierarchy that doesn’t even exist! By default, this option is not checked, allowing any files accessible by your web server software to be cloaked.

See the section “NetCloak Aliases and Caching” for information on NetCloak aliases.

Only process macros listed in “NetCloak.macros” -- Because MACRO commands may refer to files located anywhere in your web server root folder, malicious users may attempt to gain access to sensitive files on your server by uploading files or submitting data which contain such MACRO commands. To eliminate this possibility, you can check this option. When checked, you have complete control over which macros may be executed on your server -- only those macros entered into the “NetCloak.macros” file will be allowed to run.

If NetCloak is unable to insert a macro because this option is checked, it will insert an error message enclosed in an HTML comment in place of the MACRO command when the document is served.

Prohibit EXEC_CGI tags -- This option is fairly straightforward. When checked, NetCloak will not process EXEC_CGI commands. The EXEC_CGI command poses a security risk because malicious users may attempt to upload CGI files to your server and then call these CGIs by uploading HTML files containing EXEC_CGI commands. Or they may attempt to use your existing CGIs in a destructive manner. By default, this option is checked. To enable EXEC_CGI commands on your server, you must un-check this option.

If this option is checked, NetCloak will insert an error message enclosed in an HTML comment in place of the EXEC_CGI command when the document is served.

Forms Processing

Convert angle brackets to html entities— This checkbox determines whether or not articles containing the less-than (<) and greater-than (>) brackets should be converted into their HTML encoded equivalents for use on the Web. HTML reserves a small number of ASCII characters for use as formatting instructions, including the (<) and (>) brackets, and if these characters are to be used in the body of an HTML document they must be first converted. Checking the “Convert angle brackets...” option will convert these brackets into their HTML character tag equivalents, “<” (for less-than, '<') and “>” (for greater-than, '>'). When checked, articles containing greater than and less than symbols will not produce unexpected formatting when they are submitted and converted to HTML documents on your server.

When not checked, NetCloak Pro will simply leave the brackets in the user-entered text when it is inserted. This allows users to enter HTML tags and markup their articles.

The security advantage of selecting this option is that you will be able to prevent authors from embedding HTML tags in their articles, including formatting commands, images, and links to other pages. The downside, of course, is that if this option is selected, users won't have the option of entering HTML tags on their own to enhance the formatting of their pages.

Restrict access to root folder— When this checkbox is checked, all FDML activity is restricted to the configured Root Folder. No file outside this folder can be opened, read from, written to, or created in any way whatsoever. This rule is applied to all FDML primary and supplemental directives, so that, for instance, the COPY command cannot be used to save files to other mounted volumes.

This option is enabled by default, and there is rarely a reason to turn it off. Older versions of NetForms did not support Mac OS alias resolution, so allowing access outside the Root Folder provided a means of accessing other volumes. Now, however, you can access any folder on any mounted volume simply by placing an alias of the folder inside the Root Folder.

On the other hand, there are very good reasons for leaving it on. When you permit NetCloak Pro to access files outside the Root Folder,

CONFIGURING NETCLOAK

any file, even those within the System Folder, can be opened, read, or overwritten using the appropriate FDML commands. This is particularly dangerous if you provide FTP upload abilities to your users, or if you disable some of the other security settings described below. In such a situation, a malicious user with knowledge of FDML syntax could upload or submit an FDML file which contained directives instructing NetCloak Pro to overwrite your System or Finder files with meaningless garbage— thus quickly turning your server into an expensive paperweight.

Prohibit fdml tags in form fields— When this option is checked, then NetCloak Pro pre-screens all input form data and rejects any post that contains any FDML tags.

Again, this setting exists to prevent malicious users with knowledge of FDML syntax from submitting data that creates a new FDML file on your server, which could be written to return the contents of sensitive files via the user's Web browser.

FDML files must have suffix— This security setting causes NetCloak Pro to double-check the file suffix of FDML files before processing their contents. If the suffix of the file does not match the configured value (which defaults to ".fdml" and rarely need to be changed), then NetCloak Pro merely returns an error message to the user who submitted the form.

Early version of NetForms would process any file containing FDML commands which was specified in a form's ACTION attribute. This posed a potential security risk because a hacker could enter FDML commands into documents saved with an ".html" or ".txt" extension, and NetForms could then be used to retrieve files from the web server using that new, bogus, FDML file. Enabling this option causes NetCloak Pro to reject any FDML file not ending with the configured suffix. By default, the suffix is set to ".fdml". Obviously, the configured suffix should *not* be used as the suffix of any files created via CREATEDOC or TEXTSTORE directives.

Form and FDML must be on same server— When you enable this security setting, you are preventing other web sites from hijacking your form.

Because the URL that defines the location of your FDML file, such as "http://your.server.com/Recipes/Recipe.FDML", can be accessed from

anywhere on the Internet, any web site could duplicate or copy the HTML form which provides user input to your FDML. Then, anyone using that form on the other web site would submit data to your server to be processed by NetCloak Pro. This is known as hijacking your form.

This can cause many undesirable effects, such as skewing survey data collected via the form, or overloading your web server with more traffic than it was designed to withstand.

When this option is enabled, NetCloak Pro verifies that the HTML form used to submit data and the FDML file which will process the data reside on the same machine. If they do not, an error message is returned to the web browser and the form data is not processed.

Configuration Outside Of NetCloak

There are a few configuration items outside of NetCloak that have an effect on how some NetCloak commands behave. These affect DOMAIN command parameters and how NetCloak commands display dates and times.

Web Server DNS Option

Most web servers support an option to look up the domain name corresponding to the client IP address for each connection. Such an option is usually named something like “Look up client names”, “Perform DNS lookups” or “Use DNS”. This option affects how the server passes the client address to NetCloak, which affects the behavior of NetCloak’s DOMAIN commands. When it is off, the server passes the TCP/IP address of the client to NetCloak without trying to resolve the machine name. When it is on, it will pass the host name instead.

The HIDE, SHOW and INSERT_DOMAIN commands in NetCloak will use the address as it is passed from the server. This means that when DNS lookup is turned on in the server, the INSERT_DOMAIN command will insert the hostname (like “www.maxum.com”) and when it is turned off it will insert the IP address (like “198.85.68.67”). Likewise, you will have to use appropriate parameters in the HIDE_DOMAIN command depending on whether NetCloak

CONFIGURING NETCLOAK

Date & Time control panel format setting

The `INSERT_TIME` and `INSERT_DATE` commands will display the time and date using the format selected in the Date & Time control panel on the server. Also, parameters to the `HIDE_DATE` and `SHOW_DATE` commands must match the short date format set in the Date & Time control panel. Make sure that the time and date formats are set according to how you would like them displayed on your pages. Since the date and time functions of NetCloak rely on the server's system clock, make sure the time and date are accurate.

EDITING COUNTERS, GROUPS AND GLOBALS

Counters are automatically created as they are used in your HTML pages, but NetCloak also provides a way to create counters manually and to adjust their values. You can also remove counters that are no longer used, which is recommended to maximize the efficiency of the counter database. In addition, you can configure counters to automatically reset at a certain interval.

To edit counters using the NetCloak CGI, choose the Variables... command from the Configuration menu. This will display the NetCloak Lists window, and the first tab is used to edit counters.

To add a new counter, type its name into the Name field, enter a starting value into the Value field, and click the Save Changes button. To remove a counter, select it in the Counter List and click the Remove Counter button.

Clicking on a counter name in the counter list will cause the Name and Value fields to be filled in with that counter's name and current value. To change the value of a counter, click on the counter in the counter list, change the value in the Value field, and click the Save Changes button.

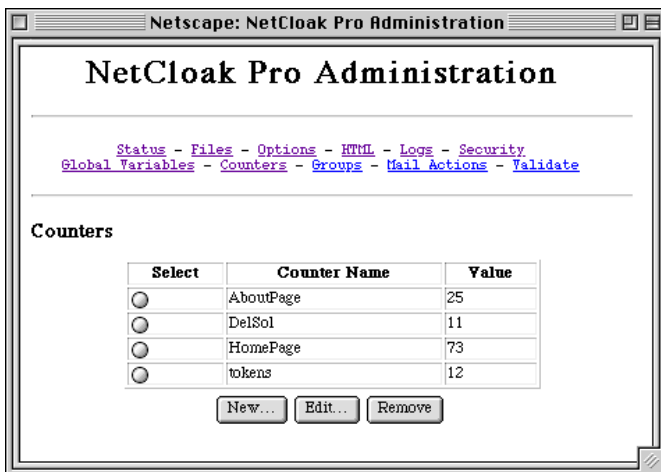


Figure 17: The Counters page of the NetCloak Web admin interface.

When using the plug-in version of NetCloak, you can use the NetCloak CGI to edit counters, as described above, but using the Web admin interface is preferred. To ensure that your updates take effect, quit

EDITING COUNTERS, GROUPS AND GLOBALS

your web server while you are editing the counters with the CGI. After your edits are complete, quit the NetCloak CGI and re-launch the server.

To edit counters using the NetCloak Web admin interface, go to the admin URL in your Web browser, then click the “Counters” link. This will display the Counters admin page, containing a table of all the currently defined counters.

To add a new counter, click the New... button at the bottom of the page. To edit the name, value, or auto-reset options of an existing counter, click the radio button next to the counter in the table, then click the Edit... button at the bottom of the page. In either case, you are then presented with the Edit Counter web admin page.

To change the name or value of the counter, change the name or value in the text fields, and click the Save Changes button.

For information on using counters within your HTML pages, see the section Commands: Counters in the Reference Guide.

Resetting Counters Automatically

Each counter you create will, by default, start with the value you assign it, and increment by one each time an `INSERT_COUNT` command is processed, forever. However, you can create counters that reset to a value you specify on an hourly, daily, weekly, or monthly basis.

To make a counter reset automatically, check the Automatically Reset Counter checkbox.

In the CGI, this will enable the Set To: and Every: fields below. After setting automatic reset options, be sure to click the Save Changes button to save the changes to the counter.

Set To: Enter the value that the resetting counter will reset to at the specified interval.

Every: Click the up and down arrows to specify whether the counter will reset on an hourly, daily, weekly, or monthly basis.

At: If the counter resets on a daily, weekly, or monthly basis, this field becomes enabled. Click the up and down arrows to specify the time of day the counter will reset.

EDITING COUNTERS, GROUPS AND GLOBALS

On: If the counter resets on a weekly or monthly basis, this field becomes enabled. Click the up and down arrows to specify what day of the week (for weekly resets) or day of the month (for monthly resets) the counter should reset on.

The screenshot shows a Netscape browser window titled "Netscape: NetCloak Pro Administration". The main heading is "NetCloak Pro Administration". Below the heading is a navigation menu with links: [Status](#) - [Files](#) - [Options](#) - [HTML](#) - [Logs](#) - [Security](#) - [Global Variables](#) - [Counters](#) - [Groups](#) - [Mail Actions](#) - [Validate](#). The "Counters" link is highlighted. Below the navigation menu is the "Edit Counter" section. It contains the following fields and options:

- Counter Name:** A text input field containing "HomePage".
- Counter Value:** A text input field containing "73".
- ☐ **Automatically reset counter**
- Set To:** A text input field.
- ☐ **Every Hour**
- ☐ **Every Day**
 - At:** A dropdown menu showing "Midnight".
- ☐ **Every Week**
 - At:** A dropdown menu showing "Midnight".
 - On:** A dropdown menu showing "Sunday".
- ☐ **Every Month**
 - At:** A dropdown menu showing "Midnight".
 - On:** A dropdown menu showing "1st".

At the bottom right of the form are two buttons: "Save Changes" and "Reset".

Figure 18: The NetCloak Web admin Edit Counter page.

In the Web admin interface, click the radio button next to the desired reset interval (every hour, every day, every week, or every month), then select the reset date/time from the popup menus underneath that radio button. Click "Save Changes" for the changes to take effect.

Because the values of counters are important to the Webmaster, NetCloak creates a file called NetCloak Counter Resets within the NetCloak Files folder where it logs the date, the time, the counter name, the current value, and the reset value every time a counter is automatically reset.

Simplifying Maintenance With Groups

There will be times when you want to use a NetCloak command and specify a series of possible criteria. For example, you may want to hide a portion of advanced HTML from early versions of Netscape, Explorer (and maybe even Mosaic) users. Or you may want to show a portion of a page only to those users that are in your internal IP domain, which spans several ranges.

To make these commands easier to use, NetCloak includes a “groups” feature, which allows you to create a group of parameters that can then be used by name within any NetCloak command. You can create and manage groups from within the NetCloak user interface, and then simply refer to the name of the group instead of each item individually in NetCloak commands on your pages. This makes on-going maintenance easier by allowing you to change the group without having to update your HTML pages. It also makes reusing lists easy anywhere on your Web site. Just define the group once and use it anywhere.

Editing groups is done on the Groups page of the NetCloak Web admin interface or the Groups tab of the NetCloak Lists window.

Editing Groups with the CGI

To edit groups from the CGI, select the Variables... command of the Configuration menu and click the Groups tab.

The example above has several groups already created, including one called SuperUsers. To see the items in any group, simply click on it and the Items list will change to a list of the items in the selected group.

Removing a single item or an entire group is as simple as selecting the item or group and clicking the appropriate Remove button.

To create a new group, enter the name of the group into the text entry field underneath the groups list, then click the Add Group button. The currently selected group or item has no effect when adding a group.

To add items to a group, first click on the name of the group in the groups list to select it. Then enter the name of the item to add to the group in the text field beneath the items list and click the Add Item button.

EDITING COUNTERS, GROUPS AND GLOBALS

Once you have created your groups, click the OK button and they will be saved to the NetCloak.groups file. Note that you can also edit this file manually with a text editor. Each line in the NetCloak.groups file represents one group.

Editing Groups with a Web Browser

You can also edit groups using the Groups page of the NetCloak Web admin interface. Enter the URL of the NetCloak Web admin page into your Web browser, then click the Groups link at the top of the page.

The screenshot shows a Netscape browser window titled "Netscape: NetCloak Pro Administration". The main heading is "NetCloak Pro Administration". Below it is a navigation menu with links: [Status](#) - [Files](#) - [Options](#) - [HTML](#) - [Logs](#) - [Security](#) - [Global Variables](#) - [Counters](#) - [Groups](#) - [Mail Actions](#) - [Validate](#). The "Groups" link is highlighted. The main content area is titled "Edit Group". It contains a "Group Name:" label followed by a text input field containing "SuperUsers". Below this are two buttons: "Save Changes" and "Reset". Underneath is the "Group Items:" section, which contains a table with two columns: "Select" and "Item". The table has four rows of data:

Select	Item
☐	john,moof
☐	steve,hair
☐	mary,123m5
☐	chris,nerd

Below the table is a section titled "Enter new item value below:" with a "New value:" label and a text input field. At the bottom of this section are three buttons: "Add Item", "Edit Item", and "Remove".

Figure 19: The NetCloak Web admin Groups page.

To remove a group and all of its items, click the radio button in the Select column next to the group's name, then click the Remove button.

To create a group, click the New... button. To edit an existing group, click the radio button next to the group's name, then click the Edit... button. In either case, you are presented with the Edit Group page.

EDITING COUNTERS, GROUPS AND GLOBALS

To change the group name, type the new name into the Group Name field, then click Save Changes.

To add a new item to the group, enter the new item into the New value field, then click the Add Item button.

The screenshot shows a web browser window titled "Netscape: NetCloak Pro Administration". The main heading is "NetCloak Pro Administration". Below it is a navigation menu with links: [Status](#) - [Files](#) - [Options](#) - [HTML](#) - [Logs](#) - [Security](#) - [Global Variables](#) - [Counters](#) - [Groups](#) - [Mail Actions](#) - [Validate](#). The "Groups" link is highlighted. The main content area is titled "Edit Group". It contains a "Group Name:" label followed by a text input field containing "SuperUsers". Below this are two buttons: "Save Changes" and "Reset". Underneath is a "Group Items:" label. Below this is a table with two columns: "Select" and "Item". The table contains four rows of items: "john_moof", "steve_hair", "mary_123m5", and "chris_nerd". Each row has a radio button in the "Select" column. Below the table is a section titled "Enter new item value below:" with a "New value:" label and a text input field. At the bottom are three buttons: "Add Item", "Edit Item", and "Remove".

Select	Item
☐	john_moof
☐	steve_hair
☐	mary_123m5
☐	chris_nerd

Enter new item value below:

New value:

Figure 20: The NetCloak Web admin Edit Group page.

To remove an item from the group, click the radio button in the Select column next to the item to be removed, then click the Remove button.

To edit an item, click the radio button next to the item, enter the new value for the item in the New value field, then click the Edit Item button.

Using Groups in NetCloak Commands

To use a group in any NetCloak command, refer to it by name within the command, preceded by an exclamation point ("!"). The exclamation point tells NetCloak to look for and use a group rather than try to use the group name as a parameter.

EDITING COUNTERS, GROUPS AND GLOBALS

Let's look at the SupportsTables example shown in the sample dialog above. Here is the HTML syntax for showing a table to certain browsers, without using groups:

```
<HIDE><SHOW_CLIENT Mozilla MSIE Mosaic>
<TABLE>
<TR><TD>...
```

By creating the SupportsTables group, we simplify this to:

```
<HIDE><SHOW_CLIENT !SupportsTables>
<TABLE>
<TR><TD>...
```

Not only does this simplify the HTML, but when a new browser comes out that supports tables, you simply add the browser to the group. No changes are needed to the HTML pages at all.

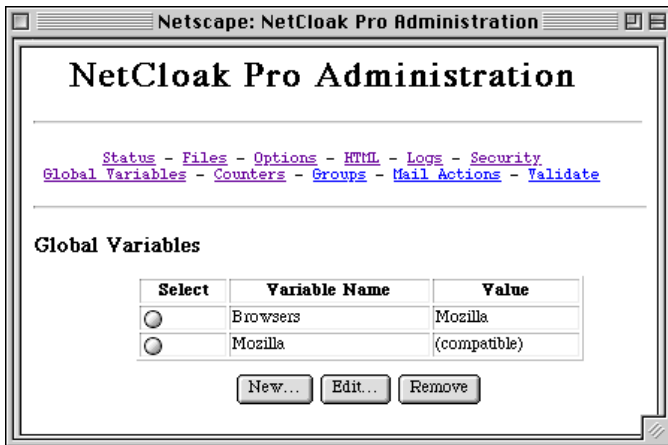


Figure 21: The Variables page of the NetCloak Web Admin.

Accessing Global Variables

Variables, as we have discussed, are available in three types: Global, Local, and User. Local and User variables last only as long as a page is being served or a user is visiting your site. Because they are relatively temporary, these variables can not be directly edited. Global variables, by contrast, are permanent, and can be set directly using the Web admin interface or the Variables window in the NetCloak CGI.

EDITING COUNTERS, GROUPS AND GLOBALS

To edit global variables using the Web admin interface, go to the NetCloak Web admin URL in your Web browser, then click the Global Variables link.

To set global variables with the CGI, select the Variables... command from the Configuration menu, and click the Variables tab.

Creating, modifying, and deleting global variables is easy. To create a variable, simply enter its name and value, and click the Save Changes button. To modify an existing variable, click on it in the Variables list, edit the value, and click Save Changes. To delete a variable, simply select it in the list and click Remove Variable.



Figure 22: The NetCloak Web admin Global Variables page.

To delete a global variable, click the radio button next to the variable's name, then click the Remove button.

To create a new global variable, click the New... button. To edit an existing global variable, click the radio button next to the variable's name, then click the Edit... button. Either way, you are shown the Edit Global Variable page.

Type changes to the variables name and/or value into the text fields, then click the Save Changes button for the change to take effect.

Global variables changed in the CGI or the Web admin pages will be immediately reflected when used in commands on your site. This is an excellent way to simplify site maintenance.

EDITING COUNTERS, GROUPS AND GLOBALS

For example, assume each of your pages used a macro for the page header. You could put something like this in the macro:

```
<HIDE><SHOW_GLOBAL MaintMode "On">  
<REDIRECT "http://mysite.com/MaintMode.html">  
<SHOW>
```

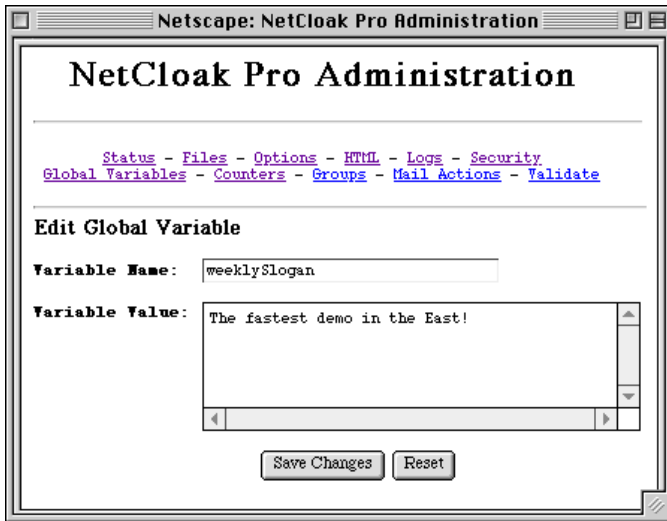


Figure 23: The NetCloak Web admin Edit Global Variable page.

"MaintMode.html" could be a page that tells visitors that the site is currently being revised and that they should check back later. Now, any time you need to make major revisions to the site, you can just edit the Global variable so that its value is "On". When you are done making changes, change the value to "Off" and the site will be accessible again.

Using AppleScript to Access Global Variables

NetCloak includes the ability to set and get global variable values using AppleScript or any application capable of sending AppleEvents.

The Get Global Event

To get the value of a global variable in the NetCloak CGI using AppleScript, use the following script syntax, replacing "varName" with the name of the global variable you want to get:

EDITING COUNTERS, GROUPS AND GLOBALS

```
tell application "NetCloak.acgi"  
    set varValue to get global "varName"  
end tell
```

This script sets the value of the script variable “varValue” to the value of the global variable. An example script using this syntax, named Get Global via NetCloak.acgi, is in the Sample Scripts folder installed with NetCloak.

Because the NetCloak Plug-In cannot publish its AppleScript dictionary when it is running inside the web server, you must use the raw AppleEvent codes in a script designed to communicate with the NetCloak Plug-In. For example, the equivalent script to talk to the NetCloak Plug-In would look like this:

```
tell application "WebSTAR 2.1.1"  
    set varValue to «event NCIkGetG» "varName"  
end tell
```

Note two differences: you must “tell” the web server, not the plug-in, what to do; and you must use the raw event code “«event NCIkGetG»”. To enter the special ‘«’ character, type “option-\” (hold down the option key and press the backslash key), and type “option-shift-\” to enter the ‘»’ character. This is necessary because the plug-in actually becomes part of the web server when it is running, but it cannot modify the web server’s AppleScript dictionary. See the Get Global via WebSTAR script in the Sample Scripts folder.

To send the Get Global event to NetCloak using some other application that requires you to enter the AppleEvent codes for the event class, event ID and event parameters, you will need to know the following information. Consult the application’s documentation for specific instructions on how to use this information.

Event Class: ‘NCIk’
Event ID: ‘GetG’

Parameters:

The direct parameter ‘----’ of the event is type ‘char’ and contains the name of the global variable.

Reply:

The direct parameter ‘----’ of the reply event is type ‘char’ and contains the variable’s value.

The Set Global Event

EDITING COUNTERS, GROUPS AND GLOBALS

To set the value of a global variable in the NetCloak CGI using AppleScript, use the following script syntax, replacing “varName” and “varValue” with the name of the global variable and the value you want to set it to:

```
tell application "NetCloak.acgi"  
    set global "varName" to "varValue"  
end tell
```

An example script using this syntax, named Set Global via NetCloak.acgi, is in the Sample Scripts folder.

Because the NetCloak Plug-In cannot publish its AppleScript dictionary when it is running inside the web server, you must use the raw AppleEvent codes in a script designed to communicate with the NetCloak Plug-In via the web server. For example, the equivalent script to talk to the NetCloak Plug-In would look like this:

```
tell application "WebSTAR 2.1.1"  
    «event NCIkSetG» "varName" ¬  
        given «class valu»: "varValue"  
end tell
```

Note two differences: you must “tell” the web server, not the plug-in, what to do; and you must use the raw event codes. See the “Set Global via WebSTAR” script in the Sample Scripts folder.

To send the Set Global event to NetCloak using some other application that requires you to enter the AppleEvent codes for the event suite, event class and event parameters, you will need to know the following information. Consult the application’s documentation for specific instructions on how to use this information.

Event Class: ‘NCIk’

Event ID: ‘SetG’

Parameters:

The direct parameter ‘----’ of the event is type ‘char’ and contains the name of the global variable.

A ‘valu’ parameter is also required; it is type ‘char’ and contains the new value for the global variable.

Reply:

There is no reply to the Set Global event.

USING THE NETCLOAK CGI

This and the Reference guide are written assuming that the NetCloak plug-in is being used. This section has more information particular to using the CGI version of NetCloak. There are important differences in installation, configuration URLs for cloaked files and other areas that are covered here.

Most Mac web servers support W*API plug-ins, and the plug-in version is recommended for servers that support it. The plug-in imposes less overhead on the server and is more convenient to use. For servers that do not support plug-ins, or special situations where the plug-in is not appropriate, use the traditional CGI version of NetCloak.

You may select either the CGI or the plug-in according to your site and server's needs, but under no circumstances should both be run simultaneously with the same server application.

Using NetCloak Without Suffix Mappings

If your server does not support action-based CGIs, you will need to add a command to your URLs to tell the web server that the NetCloak CGI must process the document. This applies to both dynamic pages and FDML templates for forms processing.

To invoke NetCloak for a particular page without a suffix mapping, simply add “/NetCloak.acgi\$” between the server name and document path in URLs. For example, the URL:

`http://www.server.com/myFolder/cool.html`

...would be cloaked if you change it to:

`http://www.server.com/NetCloak.acgi$/myFolder/cool.html`

For this to work correctly, NetCloak must reside in your server's root folder, and your server must recognize that filenames ending in “.acgi” are used to invoke CGI applications.

If you have installed NetCloak.acgi in the cgi-bin folder on your server, the URL would look like this:

`http://www.server.com/cgi-`

`bin/NetCloak.acgi$/myFolder/cool.html`

The ACTION URL of a FORM tag might look like this:

`ACTION=http://www.server.com/NetCloak.acgi$/moof.fdm1`

Note that this will direct the server to have NetCloak process the request regardless of the file suffix. Most servers also support calling a

CGI in this manner, so it is a good way to force your web server to return a page through NetCloak when you are not sure if a suffix mapping is working correctly.

This technique is unreliable because if the CGI path and \$ are left out of the URL the file will be served without being processed by NetCloak. In many cases this will only be a cosmetic problem since the NetCloak commands in the page will not work. If NetCloak dynamic page commands are being used for security this could cause a significant security hole.

Installing the NetCloak CGI

The CGI version of NetCloak is normally installed in the web server root folder or cgi-bin folder. If the NetCloak CGI application is in a folder other than the web server root folder, you will need to launch NetCloak and select the root folder on the Files tab of the NetCloak Configuration window. The NetCloak Files folder, containing NetCloak's preferences, macros and other support files must be in the same folder as the NetCloak application.

While it is possible to use NetCloak without defining an action or suffix mapping, most users configure the web server so that it calls NetCloak to process all HTML files. This will allow you to use NetCloak in any of your HTML pages without worrying about whether the page is cloaked or not.

Installing the NetCloak CGI with WebSTAR

1. Copy the NetCloak.acgi application from the NetCloak folder to the same folder as the WebSTAR application.
2. Launch WebSTAR.
3. Launch NetCloak.acgi.
4. Select “.NCLK Files to CGI” from the “Auto-Map” menu.

If you want to process all HTML files through NetCloak:

5. Select “.HTML Files to CGI” from the “Auto-Map” menu.

For forms processing with NetCloak Pro:

6. Select “.FDML Files to CGI” from the “Auto-Map” menu.

USING THE NETCLOAK CGI

Installing the NetCloak CGI with AppleShare IP

With AppleShare IP, the “NetCloak Files” folder used by the CGI is stored in the “Web Folder”.

1. Copy the “NetCloak.acgi” application from the “NetCloak 3.1” folder to the Web Folder on your AppleShare IP server.
2. Start up the AppleShare IP server, then double-click the NetCloak.acgi application to launch it. Starting NetCloak by double-clicking on it is not required, as the web server will automatically start it as needed. However, starting by hand is recommended to avoid possible start-up errors (like out of memory problems).
3. Launch AppleShare IP Admin.
4. Select “Web & FTP MIME Types...” from the Server menu.
5. Click on the line with the suffix “.html”, then click the Edit button.
6. Click on the pop-up menu for Action: and select Execute...
7. After the file open dialog box appears, select “NetCloak.acgi” in the file list, then click the Open button.
8. Click the OK button in the Edit MIME Type window, then click Save. Click the OK button in the alert that appears after you click the Save button.
9. Stop the AppleShare IP Web & File Server, then re-start it.

Create an additional MIME mapping for forms processing if you have NetCloak Pro:

1. Select “Web & FTP MIME Types...” from the Server menu.
2. Click on the line with the suffix “.html”, then click the Duplicate button.
3. Change the suffix to “.fdml”.
4. Click on the pop-up menu for Action: and select Execute...
5. After the file open dialog box appears, select “NetCloak.acgi” in the file list, then click the Open button.
6. Click the OK button in the Edit MIME Type window, then click Save. Click the OK button in the alert that appears after you click the Save button.
7. Stop the AppleShare IP Web & File Server, then re-start it.

Installing the NetCloak CGI with WebTen

1. Copy "NetCloak.acgi" from the "NetCloak 3.1" folder to the "cgi-bin" folder in the WebTen folder.
2. Double-click the "NetCloak.acgi" application to launch it. Starting NetCloak by double-clicking on it is not required, as the web server will automatically start it as needed. However, starting by hand is recommended to avoid possible start-up errors (like out of memory problems).
3. Launch WebTen.
4. Launch your web browser if it is not already running.
5. Access your server's "Administration Server" home page (at "http://www.yourserver.com/webten_admin") via your browser.
6. Click "Action Handlers".
7. In the table that appears, enter "CLOAK" in an empty cell in the first column ("Actions"), and enter "/cgi-bin/NetCloak.acgi" in the second column, "Action Handler".
8. Click "Save Handlers".
9. Click "MIME Extensions".
10. In the "User-Defined MIME Extensions" table that appears, enter ".ncl" in the first column, "text/html" in the second column, and pick "CLOAK" from the popup menu in the third column.
11. Click "Save MIME Extensions".

If you want to process all HTML files through NetCloak, also follow these steps:

12. Click "Built-In Extensions".
13. Select "CLOAK" from the "Action" popup menu for the ".html" extension.
14. Click "Save MIME Extensions".

Create a MIME extension for forms processing if you have NetCloak Pro:

1. Click "Action Handlers".
2. In the table that appears, enter "NFORMS" in an empty cell in the first column ("Actions"), and enter "/cgi-bin/NetCloak.acgi" in the second column, "Action Handler".
3. Click "Save Handlers".

USING THE NETCLOAK CGI

4. Click “MIME Extensions”.
5. In the table that appears, enter “.fdml” in the first column, “text/html” in the second column, and pick “NFORMS” from the popup menu in the third column.
6. Click “Save MIME Extensions”.

Installing NetCloak with MacOS 8 Web Sharing

Personal Web Sharing does not support WebSTAR API plug-ins, so you will only be able to use the CGI version of NetCloak.

1. Copy the “NetCloak.acgi” application from the “NetCloak 3.1” folder into the folder selected as the Web Folder in the Web Sharing control panel.
2. Double-click the NetCloak.acgi application to launch it. Starting NetCloak by double-clicking on it is not required, as the web server will automatically start it as needed. However, starting by hand is recommended to avoid possible start-up errors (like out of memory problems).
3. Run the AppleScript named Map .NCLK Files to NetCloak, located in the For Web Sharing folder in the Goodies folder installed with NetCloak.

If you want to process all HTML files through NetCloak, also follow this step:

4. Run the AppleScript named Map .HTML Files to NetCloak, located in the For Web Sharing folder in the Goodies installed with NetCloak.

Create an additional action for forms processing with NetCloak Pro:

5. Run the AppleScript named Map .FDML Files to NetCloak, located in the For Web Sharing folder in the Goodies installed with NetCloak.

Configuration

Both the CGI and plug-in have a complete web interface, but only the CGI has a traditional application interface (GUI). The application’s GUI mirror’s the web interface, with most settings accessed through

tabs in the Configuration window or other windows opened by menu commands.

The CGI application *must not be used to make configuration changes for the plug-in*. This can lead to a conflict between the CGI and the plug-in which frequently causes lost changes to configuration settings or variable or counter values.

Global Variable AppleEvents

AppleEvents for manipulating NetCloak global variables are quite different between the plug-in and CGI application. Events can be sent directly to the application instead of being sent to the web server as with the plug-in. The application is also able to publish an AppleEvent dictionary, making it possible to use very English-like names for the events.

See the section *Using AppleScript to Access Global Variables* for details on AppleEvents for global variables using either the NetCloak plug-in or CGI.

Action.....	72, 73, 74, 75, 76, 77, 78, 84, 86, 88, 89, 90, 123, 124
AppleShare IP	81, 82, 83, 85, 86, 87, 93, 123
ASCII.....	80, 101
Ask Server For Root Folder	97
BEGIN_MACRO.....	28, 29
CLIENT.....	10, 19, 26, 116
COUNT	11, 17, 26, 111
COUNTDOWN.....	10
Counter.....	10, 110, 111, 112
Counter Resets	112
Database	79
DATE.....	9, 14, 26, 58, 59, 60, 63, 66, 103, 109
Date & Time control panel.....	103, 109
DATE command.....	14, 109
Date format.....	103
DAY	23, 24
DOMAIN.....	10, 16, 22, 23, 24, 25, 109
Dynamic Caching.....	31, 103
END_MACRO.....	28, 29
Entry form.....	54
Exclude From Header	99
EXEC_CGI.....	35, 108
Extension	89
Failed Mail	46
Feedback.....	42, 62, 63, 65, 66, 68
File suffix.....	80
Footer.....	103
Form	39, 42, 45, 46, 47, 48, 57
FORM command	44, 57
Forum.....	64
From address.....	45, 46
GET	56
GLOBAL	118
Groups.....	113, 114, 115
Guestbook	47, 48, 49, 50, 51
HTML entities.....	101

HTML form.....	4, 6, 7, 39, 40, 42, 44, 48, 53, 77
HTTP header.....	35, 99
Installation.....	7, 8, 80
Internet Explorer.....	6
Latin-1.....	101
LOCAL.....	26
Log	106
Mac Roman	101
MACRO	26, 28, 29, 30, 102, 108
MacTCP.....	6
MIME extension.....	82, 89, 124
MIME mapping	82, 86, 87, 123
Mozilla	116
NETCLOAK.....	88, 89, 100
NetCloak Files folder.....	32, 46, 78, 88, 106, 112, 122
No Translation.....	101
Open Transport.....	6
Password.....	74
Path.....	37
PIXO.....	35
POST.....	44, 49, 56, 63
Post arguments	38
Prohibit EXEC_CGI tags.....	108
Quid Pro Quo.....	81, 82
Recipe	52, 54, 56
REDIRECT	19, 20, 26, 118
Relative path.....	32
Reload	30, 33, 61, 104
REPLACE_FN.....	59, 60, 63, 64, 65
REQUEST_PASSWORD.....	18, 20, 21
ResEdit.....	107
Resets, counter	111, 112
Responder	37, 38
Roman	101
Root folder	37
Send Current Date/Time.....	100
Send File's Modified Date	100
Server Name	99

INDEX

SET_LOCAL.....	26
Short Date	103
Short Time.....	103
SMTP.....	45, 77
Submit	39, 51, 56, 61, 64
Suffix, file	35, 80, 82, 89, 121
Support.....	3, 42
TagBuilder	12
Tango	36, 37, 38
Technical Support	3
THISURL.....	26
TIME.....	9, 14, 15, 24, 103, 109
Uninstalling	84, 90
USERNAME	18, 25, 26
Validate	83, 84, 91, 92
VARIABLE.....	26
W*API	121
Web Sharing	81, 82, 83, 90, 125
WebSTAR.....	4, 36, 38, 81, 82, 84, 85, 89, 90, 96, 97, 99 106, 119, 120, 122, 125
WebTen	81, 82, 83, 88, 93, 96, 124



Maxum Development Corporation

Post Office Box 315

Crystal Lake, IL 60039-0315

Printed in the USA

Copyright ©1996-2000, Maxum Development Corporation. All Rights Reserved