Index

Document:	**9000LM002EN140**
Title:	**methodica LogzMon, User Guide**
Copyright ©	**2007 methodica herger** (www.methodica.ch)

Introduction

What it is

Methodica ***LogzMon*** is a log file viewer and a simple log file monitor.
The log files are scanned and read from the web server. The results are submitted to the client where just a standard browser is needed.

Main features:

- List current log files, archived files, and any ASCII text files that can be read on the same web server machine, where LogzMon is installed.
- Show the contents of selected files or archives.
- Highlight entries matching certain criteria (REGEX[1].patterns).
- Monitor an active log file, auto-refreshing the contents.
  In Auto-refresh mode, send messages to Syslog daemons, via e-mail or SMS when log entries match certain patterns.
- Limit the amount of memory used by the browser by limiting the number of lines interpreted.
- Navigate thru pages.
- Select among predefined window sizes, typical for notebook and desktop screens.
- Multi-language support (currently English and German are implemented).

How it works

Methodica ***LogzMon*** consists of server-side and client-side components.

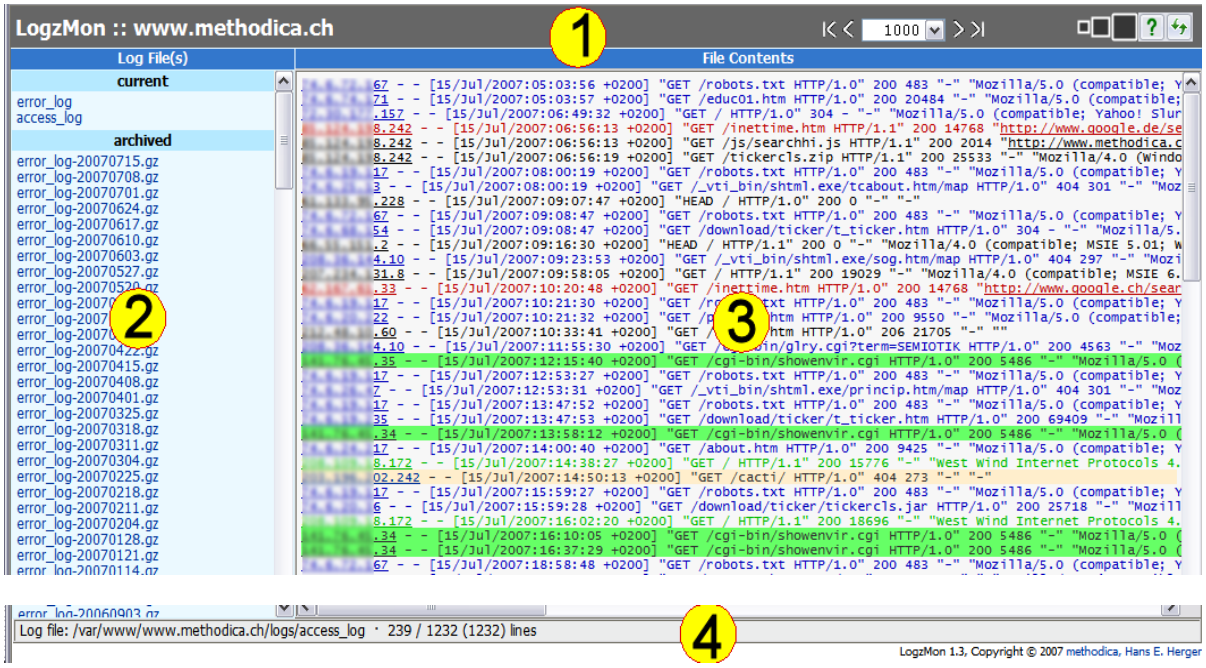
The server-side components are PHP[2] scripts, templates, and configuration files.

The client side component is dynamically generated from a template and updated by the server-side main script. It is an HTML page containing HTML code, style sheets (CSS[3]), and JavaScript code. This page should be correctly processed by modern standard browsers[4].

[1] REGular EXpression,. In the Unix / Linux world, standard format for search patterns.
[2] PHP: Hypertext Preprocessor
[3] CSS: Cascading Style Sheets
[4] LogzMon has been tested with Internet Explorer 6.0/7.0, Mozilla 1.7+, Firefox 1.5/2.0, Opera 8+.

Client and server components communicate using AJAX technology. I.e. your browser must be configured to allow for running JavaScript and, when you are using the Internet Explorer browser, the standard AJAX[5] ActiveX component provided by Microsoft.

The Client Component



The Client Elements

1: Title and Configuration bar,
2: Index Window (titled “Log File(s)”),
3: File Window (titled “File Contents”), and
4: Status Bar.

The Index Window

The index window lists the current log and archive files found on the server.
You choose a file or archive to be displayed in the file window by clicking on its name. Archives (such as ZIP) may contain sub-entries. When the archive name is clicked, the sub-entries are listed in detail, and you can choose them for display like other files.

The File Window

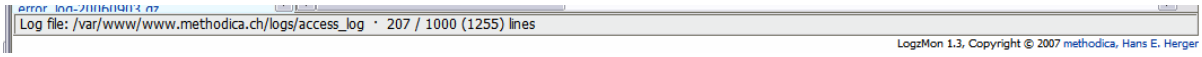
The file window shows (potentially filtered) lines of the file you choose in the index window. Lines may be hidden or highlighted (colored) according to patterns matching the log entry contents.

The Status Bar

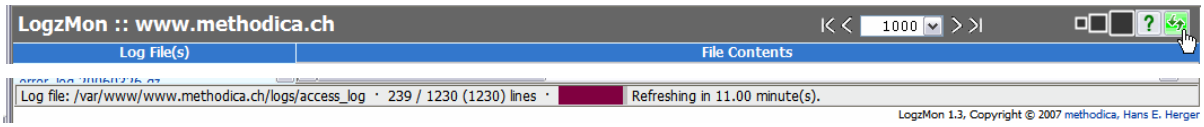
While the log files and archives are scanned or the content of a file is loaded the status bar shows the message above.

[5] AJAX: Asynchronous JavaScript and XML (although no XMLis really used by LogzMon).

When the client page is first loaded the status bar shows the number of current log files found, the number of archive files, and the number of log files contained within these archives (ZIP archives only).



Once a file's contents is loaded the status bar shows the number of lines displayed, the number range of the lines processed (see the Window Navigator tool), and the total number of lines in the file or archive (in brackets).



If LogzMon is run in *Auto-refresh* mode, additionally the remaining time in minutes - until the next refreshing will occur - is show textually and graphically.

The Tile and Configuration Bar

This bar consists of

- The Tool name ("LogzMon"),
- The Host name ("www.methodica.ch" in the example shown),
- The Navigator tool,
- The Window Sizer tool,
- The Help button
- The Auto-refresh button.

The Tool name

By clicking on the tool name a check for availability of a newer version of LogzMon is performed. If a newer version is available you are directed to a page where it can be downloaded.
For this feature an internet connection is necessary.

The Host name

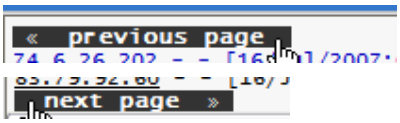
The name of the host LogzMon is monitoring. This value is defined by the configuration file. Usually you are directed to the corresponding web site if you click the Host name.

The Navigator tool

The navigator tool allows for limiting the number of lines processed per request and to navigate thru pages.
When all the lines of a log (or archive) file would be transferred to the client, a lack of memory on the client side could result. To avoid this situation, limiting the number of lines processed on the server side would be a good idea.
A drop-down list lets you choose a value between 250 and 75000, and ALL lines. Default is 1000 lines.

- The navigator buttons are (from left to right):
- Go to the first page,
- Go to the previous page,
- Go to the next page,
- Go to the last page.

Alternatively, you may click on the links shown at the top and bottom locations within the File window as shown below:



Once you have set up a limit, and as long as you do not clear cookies, the same size will automatically be set up next time you run LogzViewer for the same host.

The Window Sizer buttons

Three sizes of the client window are pre-defined:

- **Small**:
  Probably best size on most notebooks. The fonts are somewhat smaller.
- **Medium**:
  Standard size. Average size, suited for most desktops.
- **Large**:
  For desktops with sufficient window sizes only.
  On most desktops this size is comparable to a full-screen view.

Once you have chosen a size, and as long as you do not clear cookies, the same size will automatically be set up next time you run LogzViewer for the same host.

The Help button

Displays a brief online user help window.

The Auto-refresh button

This button lets you toggle the *Auto-refresh* mode ON and OFF.
It has a green background if Auto-refresh mode is ON.
In *Auto-refresh* mode, the client will issue a request to refresh the *File Contents* window every 12 minutes (720 Seconds; this value can be changed in the configuration file).
Actions (see the Configuration File below) are honored in Auto-refresh mode only.

The Configuration File

The configuration file is stored on the server as "*logviewer_config.inc.php*". It contains PHP code that is executed at runtime. Therefore the configuration follows the PHP syntax rules.

WARNING:
The configuration file is included and interpreted at runtime as PHP code.
Therefore, if the PHP interpreter detects one ore more syntax errors in the configuration file, it usually will not execute the LogzMon main script.
So, be very careful when editing this file!

The configuration file sections are:

- Logfiles,
- Logdirs,
- Archivedirs,
- Filters,
- Filtersets,

- Actions, and
- General purpose constants (defines).

Different configurations may be combined within the same configuration file by un-commenting the statement '*if ($user==…) { … }*' and by submitting the parameter '*user=user_name*' with the first call of LogzMon
E.g.: *https://www.yourserver.com/admin/logviewer/logviewer.php?user=ADMINISTRATOR*.

Logfiles

Purpose:

A list of filenames (including full paths) to be listed in the index window.

Syntax:

```
$logfiles = array (
      array ('full_filename','identifier'),
      array (…),
      …
);
```

Where:

full_filename	is the name of the log file to show, including the full file path. E.g.: '*F:\web\logs\syslog_catchall.txt*' or '*/var/www/www.yourhost.com/logs/syslog_catchall.txt*'
identifier	is an (optional) identifier to display together with the file name. This can be helpful if several log files with the same name but from different directories are to be monitored.

LogDirs

Purpose:

A list of directories containing log files to monitor.
Only log files with file extensions corresponding to the *logext* constant are listed.

Syntax:

```
$logdirs = array (
      array ('directory_name','identifier'),
      array (..),
      …
);
```

Where:

directory_name	is the directory path of log files to include. E.g.: '*F:\web\www.yourhost.com\logs*' or '*/var/www/www.yourhost.com/logs*' Do **not** add ending backslashes or slashes.
identifier	An (optional) identifier to display together with the file names. This can be helpful if several log files with the same name but from different directories are to be monitored.

ArchiveDirs

Purpose:

List of directories containing log archive files to monitor.
Only log files with file extensions corresponding to the *arcext* constant are listed.

Syntax:

```
$archivedirs = array (
```

```
        array ('directory_name','identifier'),
        array (…),
        …
);
```

Where:

directory_name	is the directory path of log files to include. E.g.: '*F:\web\www.yourhost.com\logs\archive*' or '*/var/www/www.yourhost.com/logs/archive*' Do **not** add ending backslashes or slashes.
identifier	An (optional) identifier to display together with the file names. This can be helpful if several log files with the same name but from different directories are to be monitored.

Filters

Purpose:

Filters are applied on every line of a log file or archive to determine…

- Highlighting of selected lines,
- actions to perform when *Auto-refresh* mode is set to ON.

Filters are only honored when no *Filterset* applies.

Syntax:

```
$filters = array (
        array ('filter_regex','foreground','background','actions'),
        array (…),
        …
);
```

Where:

filter_regex	is the filter pattern, as a regular expression. Foreground and background parameters are honored if the filter_regex matches the line string. Unix-/ Linux users should be quite accustomed to regular expressions. Additional information can be found here: *http://www.regular-expressions.info* *http://www.amk.ca/python/howto/regex/*. *http://en.wikipedia.org/wiki/Regular_expression* Meta characters must be escaped by preceding backslashes (\) to be taken literally. This is a comprehensive list of the meta characters: ^ $ * + ? {} [ ] \ | ( ) Patterns may be concatenated using the pipe (|) character. Instead of a pattern, the name of a constant (without apostrophes) may be given.
foreground	Either one of: • The RGB code of the foreground color, • the keyword '*hide*', or • empty string (= no highlighting). If *foreground* = '*hide*' the matching lines are not displayed at all. Otherwise the matching lines are displayed with the foreground color specified. RGB color format is: #rrggbb, where • rr = hexadecimal RED value (00..FF), • gg = hexadecimal GREEN value (00..FF), • bb = hexadecimal BLUE value (00..FF).

background	Either one of: • The RGB code of the background color, • empty string (= no highlighting).
actions	The name of one or more actions to perform when in *Auto-refresh* mode. Action names may be concatenated using the pipe (|) character.

Filtersets

Purpose:

Filters are applied on every line of a log file or archive to determine…

- highlighting of selected lines,
- actions to perform when *Auto-refresh* mode is set to ON.

The filters of a filter set are applied to lines of a file or archive if the filterset_regex matches the file or archive name only.

Only the filters of the first matching filter set are applied.

Syntax:

```
$filterset = array (
   array ('filterset_regex',
      array ('filter_regex','foreground','background','actions'),
      array (…),
   ),
   array ('filterset_regex',
      array ('filter_regex','foreground','background','actions'),
      array (…),
   ),
);
```

Where:

filterset_regex	is the filter pattern, as a regular expression, used to select the files to apply the filter rules upon. Meta characters must be escaped by preceding backslashes (\) to be taken literally. This is a comprehensive list of the meta characters: ^ $ * + ? {} [ ] \ | ( ) Patterns may be concatenated using the pipe (|) character.
filter_regex …	Filter rule as described above, see *Filters*.
actions	The name of one or more actions to perform when in *Auto-refresh* mode. Action names may be concatenated using the pipe (|) character.

Actions

Purpose:

Actions are performed when the *Auto-refresh* mode is set to ON, and the corresponding *filter_regex* (see *Filters* and *Filtersets*) matches a line string.

An action sends a message to an addressee according to one of three action types:

- SYSLOG: Sends a message to a syslog daemon within the intranet or thru the internet.
- SMTPMAIL: Send a message to an e-mail box using an SMTP server,
- SMSCLICK: Send a short message to a mobile phone using Clickatell's service. (Note: You must have an account and a HTTP API number from Clickatell (http://www.clickatell.com) to use this action type.)

Action types are implemented as add-on modules. So, it would be quite easy to add more types.

Syntax:

```
$actions = array (
     array ('action_name','action_type',
            'action_parameters',
```

```
            'message_regex',
            'message_text'
      ),
      Array(…),
);
```

Where:

action_name is a name of your choice to link filters to.

action_type *is one of the three types SYSLOG, SMTPMAIL, or SMSCLICK as described above.*

action_parameters

are parameters used by the add-on module. They are different for every module. The parameters are concatenated using the pipe character (|).

SYSLOG: Format:

```
'facility|level|host|process|daemon_ip|daemon_port'
```

facility: Facility code. E.g. 23 (= local 7).
level: Log level. E.g.: 5 (= notice).
host: Host name or IP.
process: Process name.
daemon_ip: IP address of the syslog daemon.
daemon_port: Port number where the syslog daemon is listening.
(The SYSLOG add-on always uses the UDP protocol.)

Facilities

Facility name	**Alternative Name**	**Value**
Kernel	Kern	0
User		1
Mail		2
Daemon		3
Auth		4
Syslog		5
Lpr		6
News		7
UUCP		8
Cron		9
System0	Security	10
System1	FTP	11
System2	NTP	12
System3	Logaudit	13
System4	Logalert	14
System5	Clock	15
Local0		16
Local1		17
Local2		18
Local3		19
Local4		20
Local5		21
Local6		22
Local7		23

Levels

Level name	Alternative name	Value
Emergency	Emerg	0
Alert		1
Critical	Crit	2
Error	Err	3
Warning	Warn	4
Notice		5
Information	Info	6
Debug		7

SMTPMAIL: Format:
`'from_name|from_address|to_name|to_address|subject'`

from_name: Sender's name.
E.g. `'LogzMon'.`
from_address: Sender's mailbox address.
E.g.: `'ogzmon@yourhost.com'`
to_name: Recipient's name.
to_address: Recipient's mailbox address.
Subject: E-mail subject.
E.g.: [LogzMon] Alert

SMSCLICK: Format:
`'sender|recipient'`

sender: Sender identification.
E.g. `'LogzMon Yourhost.com'.`
recipient: Recipient's mobile phone number.
E.g.: +491079999999.

message_regex Regex-pattern to match the line string.
E.g.:
`'\A([\d]{1,3}\.[\d]{1,3}\.[\d]{1,3}\.[\d]{1,3}) (.*) 401 401'`
…would match an IP address followed by some text plus the return codes 401 401. Two text particles, $1 and $2 (corresponding to the two expressions in brackets), can be referenced afterwards - $1 being the IP address and $2 the text between the IP address and the return codes.
If you don't need text particles just specify this parameter as
`'.*'`
…meaning 'any text'.
Meta characters must be escaped by preceding backslashes (\) to be taken literally.
This is a comprehensive list of the meta characters: ^ $ * + ? {} [ ] \ | ( )
Patterns may be concatenated using the pipe (|) character.

message_text Message text, optionally containing text particle references as $1,$2…$n respectively.
E.g.:
`'Access requested to protected area from IP=$1.\n\n$line'`
Note: these standard text particles are always available:

- $0: the entire matching string,
- $line: the entire line string,
- $file: the file name string.

Defines

Purpose:

Define constants.
Most of the defines are mandatory, few optional.

Syntax:

```
Define ('constant_name', 'constant_value');
```

Where:
constant_name is one of these:

logext	(Current) log file name extensions, delimited by underlines. Used together with the '*Logdirs*' section definitions. Should files without extension be included you can write two unerlines (__). E.g.: `define('logext', '_log_lg_');` Mandatory.
arcext	Archive file name extensions, delimited by underlines. Used together with the '*Archivedirs*' section definitions. Should files without extension be included you can write two unerlines (__). E.g.: `define('arcext', '_gz_');` Mandatory.
language	2-digit language code. Currently valid values are: `'en'=English, 'de'=German` Mandatory.
refresh_interval_seconds	In *Auto-refresh* mode, interval between requests to refresh the file contents in seconds (e.g. 720 = 12 minutes). Mandatory.
refresh_test_milliseonds	In *Auto-refresh* mode, interval for testing the refresh counter. Minimum interval in milliseconds the refresh-counter is updated. (E.g. 30000 = ½ minute). Mandatory.
size	Client window size preferred. Valid values are: `small, medium=standard, large` The user may change this value on the client side. Mandatory.
host_name	Name of the host you are monitoring with LogzMon. Mandatory.
host_name_url	Hyperlink to use when the user clicks on the Host name. Mandatory.
SuppressIpLookups	If a log or archive file line does contain an IP address a hyperlink is added to do lookup page[6] to identify the owner of this IP. You can suppress this feature by assigning then value *true* (without quote characters) to this constant. Mandatory.

[6] Thanks to www.domaintools.com for this feature!

SuppressHttpLookups	If a log or archive file line contains an already resolved hostname instead of an IP address a hyperlink is added to a lookup page[6] to identify the owner of this hostname. You can suppress this feature by assigning then value *true* (without quote characters) to this constant. Mandatory.
any other	For use with *filter, filtersets,* or other *configuration definitions* of your choice. Make sure you don't use quote characters when referencing such constants. Optional.