Programming utilities for Microsoft Windows's event log

gene m. stover

created Wednesday, 2006 September 13
updated Sunday, 2006 December 24

Contents

1 What is this? 1

2 License 2

3 Examples 2

4 User's manual 4
4.1 insert-log 4
4.2 print-log 5

5 Obtaining 7
5.1 Executables 7
5.2 Sources & executables 7

6 Notes about the code 9

A The `tail` program 9

1 What is this?

Two problems with the Window event log are:

1. It's not a text file.

2. Command line programs for accessing it are few (or non-existent).

I've written programs which solve these problems:

1. `print-log` dumps the events from the Windows event log. You choose the output format. You can choose tabular (CSV), Lisp, or XML.

2. `insert-log` allows you to stuff a message into the Windows event log. It's analogous to the `http://developer.ezaurus.com/slj/doc/commands/initlog.asp`[1] program on unix[2].

These programs are open source. This essay tells where to get them & how to use them.

2 License

One of the source code files, `getopt.c`, is in the public domain. I downloaded it from the web site of the TEX User's Group[3].

All other files, both source & executable, are copyrighted by Gene Michael Stover & released under the terms of the GNU General Public License [?]. Here's a copy of the copyright notice & license agreement at the beginning of each source file:

> *Copyright (c) 2006 Gene Michael Stover. All rights reserved.*
>
> *This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.*
>
> *This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.*
>
> *You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA*

A copy of the GNU General Public License is in the file `COPYING`[4].

3 Examples

The full user's manual for `print-log` is in Section 4.

Here are some examples of using `print-log`.

[1] `initlog`

[2] *unix* implies Unix, POSIX, AIX, HP/UX, NeXT STEP, Mac on Mach (whatever it's called). Oh! And Linux.

[3] http://tug.org/

[4] http://cybertiggyr.com/gene/wel/COPYING

Assume the Windows event log contains two items. The first item, inserted at noon, says "12 o'clock & all's well." The second item, inserted an hour later, says "Superstition brings bad luck.".

If you run on this event log, "`print-log -c,`", it prints:

```
; RecordNumber,TimeGenerated,TimeWritten,EventID,EventType,EventCategory,ClosingRecordNumber
28463,2006-Aug-31T12:00:00,2006-Aug-31T12:00:00,101,EVENTLOG_INFORMATION_TYPE,1,0,(concat) 12 o'cl
28464,2006-Aug-31T13:00:00,2006-Aug-31T13:00:00,101,EVENTLOG_INFORMATION_TYPE,1,0,(concat) Superst
```

The first line, which begins with a semicolon (`;`) because it's effectively a comment, shows the names of the fields.

Each of the other lines is an event log record. They are sorted from oldest to newest. As you can see from the last field of the second & third lines, our two event log records are in this output.

The "(concat)" means that `print-log` was unable to coax the Windows function http://msdn.microsoft.com/library/default.asp?url=/library/en-us/debug/base/formatmessage.asp[5] to apply a format string to the event log message's string arguments, so `print-log` is showing you the concatenation of those string arguments. This is common; in fact, it may be the way that event log records are meant to be viewed. I'm not sure at this time, so `print-log` prints "(concat)" as a reminder. I might remove that later.

Back to our example of two event log messages, if you run "`print-log -l`" on the event log of two messages, you get:

```
(EVENTLOGRECORD
 (RecordNumber . 28463) (TimeGenerated . "2006-Aug-31T12:00:00")
 (TimeWritten . "2006-Aug-31T12:00:00") (EventID . 101)
 (EventType . EVENTLOG_INFORMATION_TYPE) (EventCategory . 1)
 (ClosingRecordNumber . 0)
 (MessageString .
   "(concat) success Scheduler launched Automatic LiveUpdate."))
(EVENTLOGRECORD
 (RecordNumber . 28464) (TimeGenerated . "2006-Aug-31T13:00:00")
 (TimeWritten . "2006-Aug-31T13:00:00") (EventID . 101)
 (EventType . EVENTLOG_INFORMATION_TYPE) (EventCategory . 1)
 (ClosingRecordNumber . 0)
 (MessageString .
   "(concat) Superstition brings bad luck."))
```

Again, each event log record is shown in a single item, but this time, each item is Lisp data. Each item is a list whose FIRST is the symbol EVENTLOGRECORD & whose REST is an http://www.lisp.org/HyperSpec/Body/fun_assoccm_a_assoc-if-not.html[6] .

Finally, you could run "`print-log -x`" on the same event log of two messages, & you would see:

[5] `FormatMessage`

[6] `association list`

```
<EVENTLOGRECORD>
  <RecordNumber>28463</RecordNumber>
  <TimeGenerated>2006-Aug-31T12:00:00</TimeGenerated>
  <TimeWritten>2006-Aug-31T12:00:00</TimeWritten>
  <EventID>101</EventID>
  <EventType>EVENTLOG_INFORMATION_TYPE</EventType>
  <EventCategory>1</EventCategory>
  <ClosingRecordNumber>0</ClosingRecordNumber>
  <MessageString>(concat) 12 o'clock &amp; all's well.</MessageString>
</EVENTLOGRECORD>
<EVENTLOGRECORD>
  <RecordNumber>28464</RecordNumber>
  <TimeGenerated>2006-Aug-31T21:20:58</TimeGenerated>
  <TimeWritten>2006-Aug-31T21:20:58</TimeWritten>
  <EventID>101</EventID>
  <EventType>EVENTLOG_INFORMATION_TYPE</EventType>
  <EventCategory>1</EventCategory>
  <ClosingRecordNumber>0</ClosingRecordNumber>
  <MessageString>(concat) Superstition brings bad luck.</MessageString>
</EVENTLOGRECORD>
```

In fact, `print-log`'s XML output puts each EVENTLOGRECORD on a single line with no characters between fields. I've broken it here for readability.

You'll notice that this output is not valid XML because it does not have a single root element. There are probably other reasons it's not good XML. The idea is that it's close enough to XML that you could add your own root wrapper element.[7]

4 User's manual

4.1 insert-log

NAME bin/insert-log.exe[8] – insert a record with a one-line message into the Windows event log

SYNOPSIS print-log `-h`

print-log `[-c` *`category`*`] [-i` *`eventId`*`] [-s` *`source`*`] [-t` *`eventType`*`]`

DESCRIPTION bin/insert-log.exe[9] is a command line program which inserts a record into the Windows Event Log[10]. It provides only simple control over the record that it inserts.

[7] To be honest, I don't have a need for `print-log`'s XML output, so I haven't even tested it.

[8] `insert-log.exe`

[9] `insert-log.exe`

[10] http://msdn2.microsoft.com/en-us/library/aa385780.aspx

When you run `insert-log`, it reads a single line from its standard input. It creates a new event log record using the line it read & the values for the command line options. It inserts that new record into the Windows event log.

OPTIONS **-c** ***category*** Specifies a numeric category for the event log record. *Category* must be an unsigned integer. The default category is 0.

-h Print brief usage instructions.

-i ***eventId*** Specify a numeric id for the event. *EventId* must be an unsigned integer. The default *eventId* is 0.

-s ***source*** Specify the source of the new event log record. *Source* must be a string, & on the Windows systems I've tried, the only string which will work is "Application". The default *source* is "Application".

-t ***eventType*** Secifies the event type. *EventType* must be one of these strings:

- `EVENTLOG_SUCCESS`
- `EVENTLOG_AUDIT_FAILURE`
- `EVENTLOG_AUDIT_SUCCESS`
- `EVENTLOG_ERROR_TYPE`
- `EVENTLOG_INFORMATION_TYPE`
- `EVENTLOG_WARNING_TYPE`

EventType is not case-sensitive. For example, "`EVENTLOG_SUCCESS`" is equivalent to `eVEntLog_sUCcess`.

The default *eventType* is `EVENTLOG_INFORMATION_TYPE`.

AUTHOR `insert-log` was created by Gene Michael Stover.

BUGS

SEE ALSO
- Windows Event Log[11]

4.2 print-log

NAME bin/print-log.exe[12] – print the messages from the Windows Event Log

SYNOPSIS **print-log** `-h`

print-log `[-f] [-c` *`tab`*`]`

print-log `[-f] -l`

print-log `[-f] -x`

[11]http://msdn2.microsoft.com/en-us/library/aa385780.aspx

[12]`print-log.exe`

DESCRIPTION bin/print-log.exe[13] is a command line program which prints the records from the Windows Event Log[14] It is sort of like the `tail` program in unix. (Appendix A.)

OPTIONS **-c *tab*** Requests output in Delimeter Separated Values form with *tab* (which must be a character) as the field separator. If you use `-c`, you must specify a character for *tab*. `-c` is incompatible with `-l` & `-x`.

-f Tells `print-log` that it should keep printing new event log messages as they arrive. Use Ctrl-C to end `print-log` when it is running in this mode. The `-f` option can be used with the `-c`, `-l`, or `-x` options. The `-f` option for `print-log` resembles the `-f` option for the `tail` program from unix.

-h Print brief usage instructions.

-l Requests output as Lisp data. `-l` is incompatible with `-c` & `-x`.

-x Requests output as XML. It's not a full XML document because it does not have a single root element; it's a sequence of `<EVENTLOGRECORD>` . . . `</EVENTLOGRECORD>` elements. There may be other ways in which it's not valid XML. The idea is that you can easily wrap this pseudo-XML output to create valid XML. `-x` is incompatible with `-c` & `-l`.

AUTHOR `print-log` was created by Gene Michael Stover.

BUGS

- On the Microsoft Windows command line, it is difficult or impossible to specify a tab ("`\t`") character, pipe ("`|`") character, or backslash ("\") character as the argument to the `-c` option.

  I cannot fix it without adding to `print-log.c` a notation for special characters. Such a notation could be confusing, & has little value (to me), so I have no plan to fix.

- Even when the field separator character is a comma ("`,`"), the Delimeter Separated Values (DSV) format which `print-log` produces is not exactly the same as the Comma Separated Values (CSV) format which Excel & other Microsoft applications can read. The difference lies in the details of quote characters & escape characters. See the " http://www.faqs.org/docs/artu/ch05s02.html#id2901882[15] " chapter from [?].

  I could fix this by changing the `-c` command line option to mean CSV & adding a "`-d` *tab*" command line option which means DSV.

- You cannot specify the DTD for the XML format. The DTD (if you can call it a DTD at all) is hard-coded.

  No plan to fix.

[13]`print-log.exe`

[14]`http://msdn.microsoft.com/library/en-us/wes/wes/windows_event_log.asp`

[15]`Data File Metaformats`

- Even with the `-f` option, `print-log` prints all the entries from the event log. Instead, it could behave more like `tail` & print only the last *N* entries from the event log.

SEE ALSO • Windows Event Log[16]

5 Obtaining

5.1 Executables

If you just want the executable files (both for Microsoft Windows), here you go:

- bin/insert-log.exe[17]
- bin/print-log.exe[18]

Drop them into a directory which is in your path, then run either of them by typing its name on the command line. Detailed user's manuals are in Section 4.

5.2 Sources & executables

If you want the source code & other files which are required to compile the programs yourself, you may download wel.zip[19] or browse them individually in the tree (below).

To build them, make sure you have all the files in the directory tree as I've shown here. (If you unpacked the wel.zip[20] archive file, you should have that.) On the command line, `cd` into the `wel` directory. Edit the `setenv.bat` file to work with your environment; you'll probably have to edit the path to the `vcvars32.bat` file. Then type "`.\setdev && .\build`". The new executable files will be in `.\bin`.

- COPYING[21]
- Makefile.w32[22]
- bin/
  - bin/insert-log.exe[23]
  - bin/print-log.exe[24]

[16]http://msdn2.microsoft.com/en-us/library/aa385780.aspx
[17]`insert-log.exe`
[18]`print-log.exe`
[19]http://cybertiggyr.com/gene/wel/wel.zip
[20]`wel.zip`
[21]`COPYING`
[22]`Makefile.w32`
[23]`insert-log.exe`
[24]`print-log.exe`

  - bin/test0000.exe[25]
- build.bat[26]
- lib/
- setdev.bat[27]
- src/
  - src/getopt.c[28]
  - src/getopt.h[29]
  - src/insert-log.c[30]
  - src/lisp.cpp[31]
  - src/lisp.h[32]
  - src/log.c[33]
  - src/log.h[34]
  - src/pl.c[35]
  - src/pl.h[36]
  - src/print-log.c[37]
  - src/tab.cpp[38]
  - src/tab.h[39]
  - src/test0000.c[40]
  - src/this.h[41]
  - src/types.h[42]
  - src/xmalloc.c[43]

[25]`test0000.exe`
[26]`build.bat`
[27]`setdev.bat`
[28]`getopt.c`
[29]`getopt.h`
[30]`insert-log.c`
[31]`lisp.cpp`
[32]`lisp.h`
[33]`log.c`
[34]`log.h`
[35]`pl.c`
[36]`pl.h`
[37]`print-log.c`
[38]`tab.cpp`
[39]`tab.h`
[40]`test0000.c`
[41]`this.h`
[42]`types.h`
[43]`xmalloc.c`

  - src/xmalloc.h[44]
  - src/xml.cpp[45]
  - src/xml.h[46]
- tmp/

6 Notes about the code

1. Doesn't use the LOG module. Should use it or lose it to simplify things.

A The `tail` program

Unix[47] has a program, called `tail` which prints some of the final contents of a file or other input stream.

If you give it the "`-f`" command line option, `tail` will print the final part of the file, & it won't exit when it reaches the end of the file. As more content is appended to the input, `tail` will print them.

For more information about `tail`, see:

- "`man tail`" at your local unix command line,[48]
- Unix Manual Page for tail[49],
- "Tail (Unix)[50]" at Wikipedia,
- or search for "`unix tail`" at your favorite web search engine.

[44]`xmalloc.h`
[45]`xml.cpp`
[46]`xml.h`
[47]. . . *and* unix-like systems, which includes POSIX, HP/UX, AIX, Minix, Ultrix, Dynix, A/UX, NeXT STEP, BSD, & yes, Linux.
[48]This is your best source of information about any unix command line program.
[49]http://www.scit.wlv.ac.uk/cgi-bin/mansec?1+tail
[50]http://en.wikipedia.org/wiki/Tail_(Unix)