Subject: SystemLog class added
Posted by mdelfede on Mon, 12 May 2008 22:04:52 GMT
View Forum Message <> Reply to Message

I've rewritten (basing it on FallingDutch one) a SystemLog class, useful to log messages from user applications.
Includes documentation; here the features:

1 - Level-based logging.
You can log messages on 1 of eight different message channels,
DEBUG, INFO, NOTICE, WARNING, ERROR, CRITICAL, ALERT and EMERGENCY
Each level is switchable on/off on the fly.
Example :

```
SysLog.EnableLevels(ERROR | WARNING | DEBUG);
SysLog(ERROR) << "an error";
SysLog(NOTICE) << "a notice";
SysLog(DEBUG) << "a debug message";
```

Here, just first and third messages are logged.

2 - SysLog global object, just needed to use SystemLog package.

3 - Streaming-like << operator.

4 - Logging can be directed to Cerr() and/or Cout() streams, to
system log file and/or to Upp application log file.
Example :

```
SysLog.EnableCout(true);
SysLog.EnableCerr(false);
SysLog.EnableSysLog(true);
SysLog.EnableUppLog(true);
```

Here log is put on both Cout() stream, System log file and Upp application log file.

Enjoy !

Ciao

Max

Subject: Re: SystemLog class added
Posted by mrjt on Thu, 15 May 2008 15:06:04 GMT

I've started using this, it's very handy.

It didn't compile on Win32 though, so I've fixed it and committed the fixes to the SVN (hope you don't mind). No changes that will effect the POSIX code path.
The only slightly dodgy one is that ERROR seems to be defined somewhere in the Windows include files and I've fixed this using a conditional #undef. Renaming the enum would be better but I didn't want the change the interface.

Subject: Re: SystemLog class added
Posted by mdelfede on Thu, 15 May 2008 16:31:00 GMT

mrjt wrote on Thu, 15 May 2008 17:06I've started using this, it's very handy.

thx

Quote:
It didn't compile on Win32 though, so I've fixed it and committed the fixes to the SVN (hope you don't mind). No changes that will effect the POSIX code path.

no problems, it's on bazaar to be used/corrected
I didn't test on windows, 'cause I mainly work on Linux.

Quote:
The only slightly dodgy one is that ERROR seems to be defined somewhere in the Windows include files and I've fixed this using a conditional #undef. Renaming the enum would be better but I didn't want the change the interface.

we could also rename the enum... I put it in an "in-class" enum to avoid name clashes with other code, but there's no limit on Microsoft bad coding style
We could also change all enum's names with just uppercase first letter.... That could solve name clashes. Feel free to change this, I'm just using it for a short time, so it's no problem to change the interface.... better now than later.

BTW, my class does have a small problem that I'd like to solve.
Using unix system log (and I guess windows too...) on each log write a newline is appended, making less useful the << operator :

SysLog(ERROR) << "a message" << "another message"

will put the messages on 2 different lines.... so I added an automatic "\n" to console output and upp log one to have a consistent behaviour.
I'd prefere to allow more logs on the same line, but to overcome the linux log limit the only solution would be line buffering... Maybe I'll try to implement such a stuff.

Ciao

Max
