



POD Translation
by *pod2pdf*

ajf@afco.demon.co.uk

Elapse.pm

Table of Contents

Elapse.pm

NAME	1
DESCRIPTION	1
SYNOPSIS	1
Usage	1
Sample Output	1
Additional example code	1
'BUGS'	2
EXPORT	2
AUTHOR	2
Author	2
Last Update	2
COPYRIGHT	2
SEE ALSO	2

NAME

Time::Elapse - Perl extension for monitoring time conveniently during tasks

DESCRIPTION

Time::Elapse is a very simple class with one method: lapse.

Basically, the lapse method 'eats the brains' of the variable, squirrels away whatever value it may have held internally, (much like space aliens are known to do in the movies), and also stores the current time within it. Then, whenever you access the value of the variable, the 'alien' within formats the time differential between when you initialized the variable, and when you printed it, and returns that (along with any value the variable may hold, as well). :-) Every time you print it, you get the updated differential, returned by the method hidden inside the variable itself. The output will be formatted as HH:MM:SS.000000 [in Microseconds].

Frankly it doesn't do much more than time(), but then again the simplest things rarely do. :-)

All it really does is hides the calculations that anyone else would have had to set up manually in a clever way and then produce a reasonably formatted output which lends itself equally well to single-line output or inlining with other text.

SYNOPSIS

Usage

To use Elapse is simplicity itself:

```

use Time::Elapse;

# somewhere in your program...
Time::Elapse->lapse(my $now);
# or you can do:
# Time::Elapse->lapse(my $now = "processing");

#...rest of program execution

print "Time Wasted: $now\n";

```

To update the description and reset the time counter mid-stream, simply assign to the variable

```
$now = "parsing";
```

somewhere in the middle of the program. The new value is stored, while the original time is replaced with the current time.

Sample Output

Output looks something like this, using above code:

```

Time Wasted: 00:00:05.565763
or
Time Wasted: 00:00:03.016700 [processing]
(more output)
Time Wasted: 00:00:02.003764 [parsing]

```

Additional example code

You can also use this during a Net::FTP download loop of files to show elapsed time for each file's download.

```

foreach my $file (@files_to_download)
{
    # extract localfile name from $file
    # ...
    Time::Elapse->lapse(my $now = "Downloading $localfile.");
    $ftp->get($file, $localfile) or carp("### Could not download $file! $!") and n
    print "Done. Elapsed : $now\n";
}

```

```
    # ...  
}
```

This can also be a useful trick when you're processing a lot of data from multiple sources.

'BUGS'

Elapse offers time granularity smaller than 1 second, but values are approximate since the accuracy is slightly hampered by the virtue of the process itself taking somewhere roughly around 0.00001 - 0.0009 seconds. (depending on the system and how many processes are running at the time. :-)

```
#!/usr/bin/perl  
use Time::Elapse;  
Time::Elapse->lapse(my $now = "testing 0");  
for (1 .. 5)  
{  
    print "$now\n";  
    $now = "testing $_";  
}  
print "$now\n";
```

(results from a PowerMac G3/400 running MacPerl 5.004
on MacOS 8.6)

```
00:00:00.000937 [testing 0]  
00:00:00.000743 [testing 1]  
00:00:00.000344 [testing 2]  
00:00:00.000327 [testing 3]  
00:00:00.000358 [testing 4]  
00:00:00.000361 [testing 5]
```

(results from an AMD Duron 1.1Ghz running Perl 5.8.0
on RedHat Linux 8.0)

```
00:00:00.000079 [testing 0]  
00:00:00.000035 [testing 1]  
00:00:00.000018 [testing 2]  
00:00:00.000016 [testing 3]  
00:00:00.000016 [testing 4]  
00:00:00.000020 [testing 5]
```

EXPORT

None by default.

AUTHOR

Author

Scott R. Godin, <mactech@webdragon.net>

Last Update

Sun Jan 19 07:01:24 EST 2003

COPYRIGHT

Copyright (c) 2001 Scott R. Godin. All rights reserved. This program is free software; you can redistribute it and/or modify it under the same terms as Perl itself.

SEE ALSO

Time::HiRes.