

Re: How do I properly use global variables?

Source: <http://coding.derkeiler.com/Archive/Perl/perl.beginners/2007-10/msg00529.html>

- *From:* mwhipple@xxxxxxxxxxxxxx (Matthew Whipple)
 - *Date:* Tue, 23 Oct 2007 04:39:14 -0400
-

Jeff Pang wrote:

On 10/23/07, monk <rsarpi@xxxxxxxx> wrote:

I'm having problems accessing a variable outside its subroutine.
I've tried several combinations too long to write here. Maybe I just can't see the forest for the trees. But I'm lost. I need your wisdom.

I'd like my program below to change \$status to zero to exit the loop.

```
meaning...$> perl test.pl --start  
it prints out indefinitely "hello world"
```

```
But if $> perl test.pl --stop  
it gets out of the loop exiting the program.
```

And if you want to keep the notation similar to that above you'd need the second instance of the script (with the `--stop`), to send the signal to the process of the first instance (generally by storing the pid in a file and then sending the signal as below if the pid is still running).

Hi,

When script is running, how can you re-run it with another argument to make it stop?

The general way to let a running program stop is to send a signal.

Let me modify your code to,

```
use strict;  
use warnings;
```

```
our $status = 1;  
$SIG{TERM} = $SIG{INT} = sub { $status = 0 };
```

Re: How do I properly use global variables?

```
start();

sub start {
while ($status){
print "hello world!\n";
sleep 1;
}

print "out of loop. Will exit now\n";
exit 0;
}

__END__
```

When you run it, you can send SIGINT or SIGTERM to let it exit gracefully. Given the process id is 1234, under unix you can say,

```
$ kill -s 2 1234
```

The script would print "out of loop. Will exit now" and exit. (-s 2 means sending SIGINT, see `man 7 signal` for details).

The most important change for the code above is that we re-defined signal handlers:
\$SIG{TERM} = \$SIG{INT} = sub { \$status = 0};

When the script receives SIGTERM or SIGINT, it sets the global \$status to 0, so the loop condition becomes false, and the program exits.