# IEPM Coding Perl Tainting

Use the taint and warning features. The first two lines of your code should look like:

1. `#!/usr/bin/perl -w`

Or if this is a CGI script then use the taint (-T) option also:

1. `#!/usr/bin/perl -wT`

Perl: Strict, Warnings, and Taint
By Thomas Gutschmidt
Taken from: http://www.developer.com/lang/perl/article.php/1478301

There are three common tools to help Perl programmers write clean and maintainable code: the strict pragma, the warnings pragma, and taint checking. Strict and Warning are probably the two most commonly used Perl pragmas, and are frequently used to catch "unsafe code." When Perl is set up to use these pragmas,
the Perl compiler will check for, issue warnings against, and disallow certain programming constructs and techniques. In Perl (5.6.0 or later), pragmas are set up with the use command:

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

The strict pragma checks for unsafe programming constructs. Strict forces a programmer to declare all variables as package or lexically scoped variables. Strict also forces specific syntax with sub, forcing the programmer to call each subroutine explicitly. The programmer also needs to use quotes around all strings, and to call each subroutine explicitly, which forces a distrust of bare words.
The warnings pragma sends warnings when the Perl compiler detects a possible typographical error and looks for potential problems. There are a number of possible warnings (check the man pages or Activestate's perldiag document page), but warnings mainly look for the most common syntax mistakes and common scripting bugs.

Perl's ability to allow unchecked data has lead to a number of problems with Web servers and cgi utilizing Perl. To assist programmers in avoiding suspect data, Perl has taint checking built in.

Taint prevents code by automatically tagging any variable assigned from outside of the program as "tainted" and therefore unsafe. Taint specifically marks user input, file input, and environment variables. With taint enabled, a program cannot use tainted data to affect anything outside of the actual script. Data that has been marked as tainted spreads the mark to any other data it comes in contact with, so if you use a tainted variable to change a second variable within the script, the second variable also becomes tainted.

Taint basically halts any data being sent through eval, system, exec, or open calls. Taint also stops you from calling any external program without first setting a PATH environment variable. To turn on taint, you need to use a -T switch:

`Perl program.pl -T`

On a Web server, taint should be added to the path the Web server uses to search for the Perl distribution. If Perl is located within the usr/local.bin folder, each Perl script on that Web server should begin with the line:

`#!usr/local/bin/perl -T`

It is normally recommended that all Perl script used online or on a Web server should have taint enabled, and that programmers always use the strict and warnings pragmas. None of these are capable of writing secure code for you, however.
Warnings will alert you to common bugs, strict will alert you to common syntax errors, and taint forces you as a programmer to think about what you are doing with outside data.

One of the RFCs (request for change) in Perl 6 is more of a request for stagnancy. RFC number 16 is to "Keep default Perl free of constraints such as warnings and strict." One of Perl's strengths is its ability to be a quick and dirty solution. Quick hacks, one liners, and fast fixes are the breath of life of Perl, but they also often break common syntactical rules that taint, warnings, and strict would detect.
This RFC will help ensure that Perl continues to be the language of choice when it comes to the often-necessary quick fix. But, by definition, quick fixes need to be quickly written; making these useful tools the default behavior would require that one code in lines to turn them off. This would be a major drawback for short, simple scripts, while larger projects would hardly suffer by being forced to include them if the programmer needs to use them.