Perl 5 Handout

CS 4173　　　Winter 2006

1 Basic Program Template

```
#!/opt/bin/perl -w
              # -w turns on some error checking
use diagnostics; # turns on verbose error messages
use strict;      # stops us from making simple mistakes

my $var;         # declare scalar variable
```

Tips

- `$\ = "\n";` will append `\n` to all output
- `perl -Tc` *`script.pl`*
  will check your script for syntax errors
- declare all variables with `my`
- have the *Perl Pocket Reference* nearby
- use the built-in `warn()` or `carp()` functions for non-fatal error messages
- use the built-in `die()` or `croak()` functions for fatal errors

2 Useful Documentation

- Perl resources subsection of the CS 4173 website
- Perl documentation (at `perldoc.com`)
- `CGI.pm` documentation (at `stein.cshl.org`)

3 Beware of Perl 4 Code

Perl 4 is a different language than perl 5. Perl 4 is unsupported and highly unsafe. Do *not* use it.

There used to be lots of perl 4 code on the 'net and in books. Be on the lookout for these indications of perl 4:

- using `include` instead of `use` to specify modules that are referenced

  This is a sure sign on perl 4

- using `cgi-lib.pl`

  The perl 4 only module `cgi-lib.pl` was replaced by `CGI.pm`

- using `local` instead of `my` to declare variables

  There are some rare cases where `local` is still needed, but we mostly use `my` now. Local variables should be declared with `my` not `local`.

- calling functions using `&`*`function`*

  This is legal in perl 5 (your textbook does it) but not very common with perl 5

4 Using CGI With Perl 5

Program Template

```
#!/opt/bin/perl -Tw
              # -T turns on taint checking from command line

use CGI::Carp 'fatalsToBrowser';
                    # error messages won't be only in log file
use CGI':standard'; # basic CGI support
use diagnostics;    # turns on verbose error messages
use strict;         # stops us from making simple mistakes
$| = 1;             # flush buffer after each output

print "Content-type: text/html\n\n";
```

Notes

- At Dal FCS, CGI programs are run using the suexec method:

  1. the `http` daemon reads the script file
     - the file must be in your `public_html/cgi-bin` directory

    - the file must have a `.cgi` extension
2. the `http` daemon forks a new process to run the script
3. that process is owned by you
    - any files that you can read, that process can read
    - any files that you can erase or change, that process can erase or change too

  At many other sites the process that executes the CGI script does not have the same permissions as the owner of the file.

- CGI output must begin with a MIME type header that specifies what type of output follows. The header must be followed by two newlines.

  You can do this in perl as simply as this:
  `print "Content-type: text/html\n\n";`

  If you use `CGI.pm` then you can simply do this: `print header;` (for HTML files) or `print header("text/html");` (for plaintext files).

Untainting

When your programs use CGI they can contain values that come from the user. Before any of values that could have come from outside your program are used for anything that could be dangerous (e.g. opening a file, running a program) you must check that the value of the variable is okay. Variables with values that come from outside your program are called *tainted variables*.

Perl forces us to check the values of such variables when it is in *taint mode*. You can turn on taint mode by have `-T` as the first argument to perl when it starts (e.g. `perl -T` ***`myscript.pl`*** from the command line). Taint mode is automatically on when you use perl with CGI.

The only way you can remove the taint from variables is to use regular expression backreferences[*]:

```
$file = param("filename");
if ($file !~ m{^([\w.-]+)$}) {
   die "filename ‘$file’ has invalid characters\n";
} else {
   $file = $1;
}
```

[*]The following example is based on one from *The Perl Cookbook* [Christiansen and Torkington, 1998, p. 68]

Debugging

- ❑ Does you script run properly for you from the command line?
- ❑ Can the `http` daemon read your script file?
  - ❑ Are the permissions right?
  - ❑ Is it in your `public_html/cgi-bin` directory?
  - ❑ Does its name end with `.cgi`?
  - ❑ Are you using
    - `http://www.cs.dal.ca/~`*`you`*`/cgi-bin/`*`yourscript.cgi`* or
    - `http://torch.dal.ca/~`*`you`*`/cgi-bin/`*`yourscript.cgi`*?

    (either is okay)
- ❑ Does your script's output begin with the correct MIME type?
- ❑ Is your script's output buffered? (It should not be, use `$|=1;` to turn output buffering off.)
- ❑ Are there any helpful error messages in `/var/log/apache/error_log` or `/var/log/apache/suexec` about your script?
- ❑ Have you gone over Tom Christensen's checklist?

References

Tom Christiansen and Nathan Torkington. *Perl Cookbook: Tips and Tricks for Perl Programmers*. O'Reilly & Associates, Inc., 1998. ISBN 1-56592-243-3.

Johan Vromans. *Perl 5 Pocket Reference*. O'Reilly & Associates, Inc., third edition, 2000. ISBN 0-596-00032-4.