CSE2395/CSE3395

Topic 10

CGI

Learning Perl 2nd edition
chapter 19, pages 180-209
`CGI` manpage

CSE2395/CSE3395

Last time

Covered in Topic 9

- Processes
  - invoking other programs with `system`
  - talking to other programs
    - backquotes
    - opening a pipe with `open`
- Formats
  - defining with `format`
  - using with `write`

CSE2395/CSE3395

To be covered today

- The World Wide Web model
  - HTTP and HTML
- The Common Gateway Interface (CGI) model
  - interactive documents on the World Wide Web
- Installing and running CGI programs
- The `CGI.pm` module
- Passing parameters to CGI programs
  - submitting forms
- Generating HTML

CSE2395/CSE3395

The World Wide Web

Origins and philosophy

- WWW started as a mechanism for linking hypertext across the Internet
  - documents contain links to other documents
- Documents were considered static and stateless
  - requesting the same document twice always returned identical copies
- Documents were primarily text
  - focus was on content, not presentation
  - contained some rudimentary markup for formatting
- Much of this has now changed

CSE2395/CSE3395

The World Wide Web

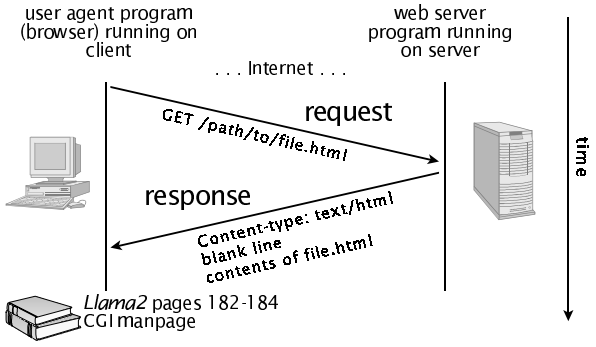
Definitions and terminology

- Documents are identified with a Universal Resource Locator (URL)
  - unique string identifying document's location
  - http://www.csse.monash.edu.au/~debbiep/cse2395/
- Documents are requested and sent using Hypertext Transfer Protocol (HTTP)
  - simple text-based file-transfer protocol understood by both ends of a transfer
  - form of responses strongly resembles email headers
- Documents are often written in Hypertext Markup Language (HTML)
  - text-based, strongly resembles rich text format (RTF)

CSE2395/CSE3395

Fetching a document

HTTP in action



Llama2 pages 182-184
CGI manpage

The CGI model

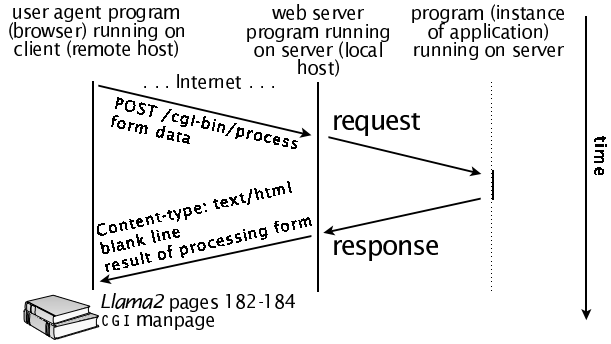
Dynamic generation of web documents

- Server may run a program (CGI program) that produces the data to be sent to the client
  - program to run is designated with a URL
- Program produces the entire response
  - including HTTP headers
- Server needs to distinguish between serving a file as-is or running it
  - two common approaches
    - run anything ending in `.cgi`
    - run anything in the `/cgi-bin` directory
  - `cgiwrap` program allows users to run CGI programs with their own access rights

CSE2395/CSE3395

The CGI model

Submitting a form



Llama2 pages 182-184
`CGI` manpage

CSE2395/CSE3395

Installing a CGI program

The setup at Monash

- Install program in ~/WWW/cgi-bin/myprogram
- Permissions must be set correctly
  - cgi-bin directory and parent directories must be searchable by others
    - `chmod +x ~ ~/WWW ~/WWW/cgi-bin`
  - program must be readable and executable by you
    - `chmod u+rx myprogram`
- Program is accessible at URL http://www-cgi.monash.edu.au/cgi-bin/cgiwrap/~you/myprogram

CSE2395/CSE3395

Example

A simple CGI program

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

# This construct is called a "here document"
# and is really just a fancy way of writing
# long strings that span several lines.
# All lines between the "<<" and the matching
# end-marker are included in the string.
# Note the HTTP header (Content-Type) and the
# blank line.  These are required.
print <<EOT;
Content-Type: text/html

<HTML><HEAD><TITLE>Hello</TITLE></HEAD>
<BODY><H1>Greetings</H1><P>Hello, world!</P>
</BODY></HTML>
EOT
```

CSE2395/CSE3395

Passing form data

Giving parameters to a CGI program

- Form data parameters are passed to CGI programs as name-value pairs
  - `species=human&language=English`
  - HTTP supports two methods
    - With GET method, parameters are attached to end of URL
      - http://www.frotz.org/cgi-bin/program?species=human&language=English
    - With POST method, parameters are included in HTTP request
- Perl comes with a standard module, CGI.pm, which makes fetching parameters easy
  - imports the `param` subroutine which can access CGI parameters

Llama2 pages 185-186
`CGI` manpage

CSE2395/CSE3395

Using CGI.pm

Accessing the module's features

- CGI.pm is normally an object-oriented module
- Simplified function-based version can be used instead
  - programmer must name functions to import
  - `use CGI ("param");`
- Can name groups of functions to import
  - `use CGI (":cgi");`
  - `:cgi` tag includes `param` and other related functions

Llama2 pages 185-187
`CGI` manpage

CSE2395/CSE3395

Example

Accessing CGI parameters

CSE2395/CSE3395

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

use CGI qw(param);

# Fetch the values of the parameters
# "species" and "language" into variables.
$kind = param("species");
$tongue = param("language");

print <<EOT;
Content-Type: text/html

<HTML><HEAD><TITLE>Greetings</TITLE></HEAD>
<BODY>
<P>Hello, $kind!  Do you speak $tongue?</P>
</BODY></HTML>
EOT
```


HTML shortcuts

Saving some typing

CSE2395/CSE3395

- CGI.pm provides HTML shortcut functions that generate HTML elements
  - with shortcut: `print h1("Heading");`
  - without shortcut: `print "<H1>Heading</H1>";`
- Shortcuts exist for most HTML elements
  - forms, tables, text markup
- Need to import the `:html2`, `:html3` and `:form` tag sets
  - `use CGI qw(:html2 :html3 :form);`

Llama2 pages 186-188
`CGI` manpage

Example

Using HTML shortcuts

CSE2395/CSE3395

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

# :standard tag set includes :html2 :form :cgi
use CGI qw(:standard);

# This prints the standard HTTP header.
print header();

# Most of these shortcuts are explained in the
# CGI manpage.  Most have names reflecting the
# HTML element.

# Start generating HTML.
print start_html("Hello"),
  h1("Greetings"),
  p("Hello, World"),
  end_html();
```


Forms

Obtaining information from the user

CSE2395/CSE3395

- HTML provides elements for fill-out forms
  - user fills out values on client (browser)
- CGI.pm provides shortcuts for generating form elements
  - text field allows user to type a single line of text
  - hidden field is not shown to user
    - cannot be changed
    - useful for saving state information between invocations
  - submit button sends completed form data to a URL
  - other button types
    - checkboxes, drop-down menus, scrolling lists, etc.

Llama2 pages 188-192
`CGI` manpage

Example

Generating a form

CSE2395/CSE3395

```
# This code generates:
# <HTML><HEAD><TITLE>Greetings</TITLE></HEAD>
# <BODY><FORM action="http://www.frotz.org/cgi-
# bin/greet"><P>What is your species?
# <INPUT type="text" name="species"></P>
# <P>What is your languange?
# <INPUT type="text" name="language" value="English"></P>
# <P><INPUT type="submit" name="Continue"></P>
# </FORM></BODY></HTML>

print header(), start_html("Greetings"),
  start_form(
    -action =>
      "http://www.frotz.org/cgi-bin/greet"),
  p("What is your species?", textfield("species")),
  p("What is your language?",
    textfield("language", "English")),
  p(submit("Continue")),
  end_form(), end_html();
```


Example

Generating and processing a form with the same program

CSE2395/CSE3395

```
use CGI (:standard);
$kind = param("species");
$tongue = param("language");

# Do the parameters exist?
unless (defined $kind && defined $tongue) {
  # Generate the form.
  print header(), start_html("Greetings"),
  # With no parameters, start_form() returns own URL.
  start_form(), p("What is your species?",
    textfield("species")),
  p("What is your language?",
    textfield("language", "English")),
  p(submit("Continue")),
  end_form(), end_html();
} else {
  # Process the form results.
  print header(), start_html("Greetings"),
  p("Hello, $kind! Do you speak $tongue?"), end_html();
}
```

CGI.pm conventions

Things to note when using the module

- With form generation shortcuts, parameter values are sticky
  - submitted parameter values are re-used in subsequent generation of the same form elements
  - use `param("name", "new value")` to change a parameter's value
- Most functions have two call formats
  - `textfield("name", "value", 20, 95)`
  - `textfield(-name => "name", -size => 20, -default => "value", -maxlength => 95)`
    - with tagged form, arguments can be listed in any order

Llama2 pages 193-194
`CGI` manpage

CSE2395/CSE3395

Further reading

Learning more about CGI and the Web

- `CGI` manpage
- *Learning Perl*, chapter 19
- *CGI Programming with Perl*
  - Scott Guelich, Shishir Gundavaram, and Gunther Birznieks, O'Reilly 2000
- *Perl Cookbook*, chapter 19
  - Tom Christansen & Nathan Torkington, O'Reilly 1998
- *HTML and XHTML: The Definitive Guide*
  - Chuck Musciano & Bill Kennedy, O'Reilly 2000

CSE2395/CSE3395

Covered today

- The WWW model
- The CGI model
- Installing and running CGI programs
- The `CGI.pm` module
- Passing parameters to CGI programs
  - the `param` function
- Generating HTML
  - with raw strings or shortcut functions

CSE2395/CSE3395

Going further

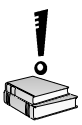
More things related to today's topic

- LWP
  - Library for the WWW with Perl
  - how to make Perl your web browser
  - on CPAN
- `HTML::Parser` and `XML::Parser`
  - tools for processing HTML or XML
  - on CPAN
- `GD`
  - module to create images on the fly
  - on CPAN
- Tainting
  - dealing with insecure data
  - *Camel3* pages 558-568; *Camel2* pages 256-363

Next time

To be covered in Topic 11

- References
  - Perl's answer to pointers
- Nested data structures
  - multi-dimensional arrays
  - emulating C `structs`



Reading:
Programming Perl 3rd edition chapters 8-9, pages 242-287
Programming Perl 2nd edition chapter 4, pages 243-275
`perlref`, `perlreftut`, `perllol`, `perldsc` manpages

CSE2395/CSE3395

CSE2395/CSE3395