CS3157: Advanced Programming

Lecture #5

Feb 6

Shlomo Hershkop

shlomo@cs.columbia.edu

Outline

- CGI
- File handling issues
- Stuff
- CGI security
- CGI Graphics
- Alternative Technologies
- Threading

- Reading: chapter 8
- For next week chapter 9

Announcement

- New TA
  - Ankur Khanna
  - OH: Thursday 10:30am-12:30pm
- Survived lab1
  - Hopefully you feel comfortable with perl and basic file manipulations
  - If you are having homework problem, please remember ..OH
  - Hope to have grading back by Wednesday’s lab, I will answer any questions about “how” to do something at the lab (or OH).

Reminder

- Lab is on CS system
- A confirmation is sent to your CS account

1. Either read your cs mail (pine)
2. Create a “.forward” file with your cunix email in your cs home dir, and it will automatically be sent to you cunix account (do a test to be sure).

From last class

- When web server executes your perl script, the %ENV is the array specific values are set with status information
  - Can get person’s IP
  - Can pass information to your script
- Input/Output is redirected for your automatically
  - Output of your script to webserver

Simple example

- http://www.cs.columbia.edu/~name/a.pl

- User in browser invokes perl script
- Web server calls script
- Perl script runs and print out a html code
- Web browser renders the webpage

Next step

- Not just execute the script want to get some starting information from the user

Forms

- One way to get information is to collect data
  - Registration
  - Payment
  - Surveys
- Commands
  - Possible choice combination
  - Actions
- Generally user needs to hit submit for anything to happen

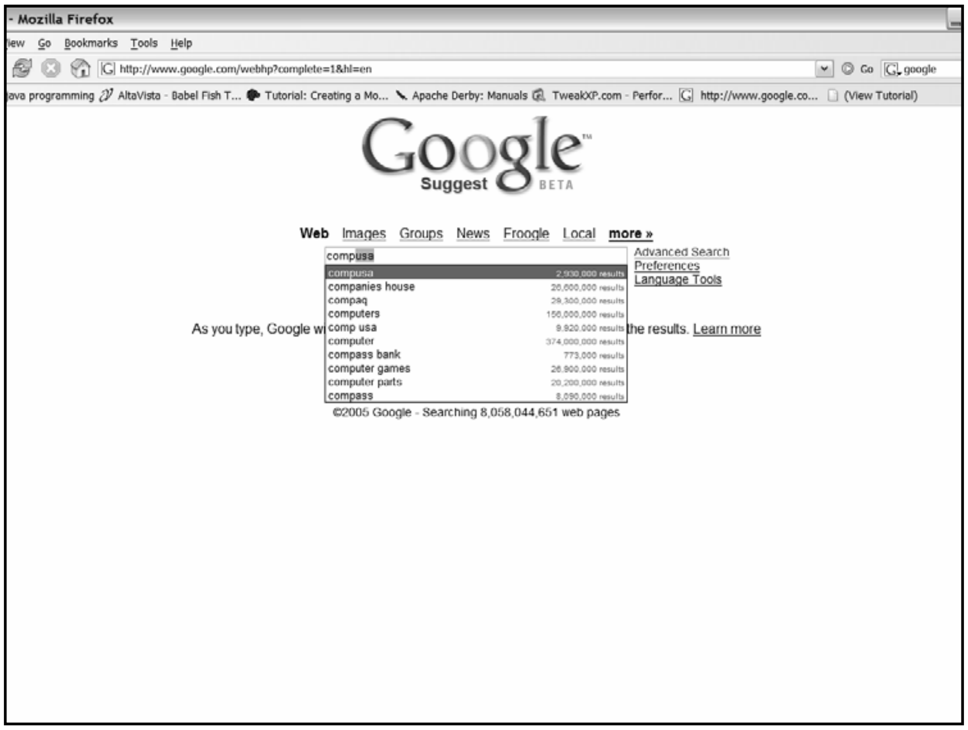
Example

- Google.com

- Load page
- Do nothing…nothing happens
- Type search…nothing happens
  - Hit submit/return trigger action

Other way

- React to user typing (will not be doing this)


- Mozilla Firefox
iew Go Bookmarks Tools Help
http://www.google.com/webhp?complete=1&hl=en
Go
google
Java programming AltaVista - Babel Fish T... Tutorial: Creating a Mo... Apache Derby: Manuals TweakXP.com - Perfor... http://www.google.co... (View Tutorial)
Google Suggest BETA
Web Images Groups News Froogle Local more »
Advanced Search
Preferences
Language Tools

compusa

compusa	2,930,000 results
companies house	26,600,000 results
compaq	29,300,000 results
computers	156,000,000 results
comp usa	9,920,000 results
computer	374,000,000 results
compass bank	773,000 results
computer games	26,900,000 results
computer parts	20,200,000 results
compass	8,090,000 results

As you type, Google wi the results. Learn more

©2005 Google - Searching 8,058,044,651 web pages

2 ways to do it

1. Create a HTML file and display a form, and your script gets input from the form
2. Have your script run
    1. If no information is being passed, print out the html for a form (then end)
    2. Else process the form information in the script

Interacting

- GET
  - HTTP request directly to the cgi script by appending the URL
- POST
  - HTTP request in content of message, i.e it is stdin to your script
- Format of GET (default):
  - Value=key separated by &
  - Space replaced by +
  - URL conversion characters

Input Tag

- Each field is in an input tag
- Type
  - Text
  - Radio button
  - Checkbox
  - Pull down menus
  - etc
- Name
  - Symbolic name (so can recognize it)
- Value
  - Default value, or what the user will end up typing

Encoding

- Spaces are turned to +
- & separates field
- Special characters are turned into %?? (hex)
  - “(“ is %28

  - So “class is great” = “class+is+great”

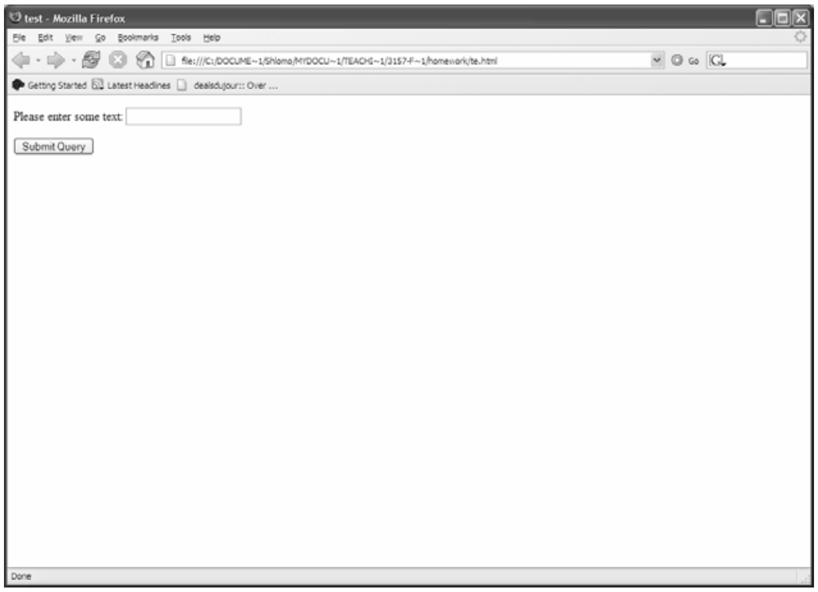
others

- Submit buttons
  - <input type=“submit”>
- Reset buttons
  - <input type=“reset”>

- Value will change the default name on the button

Putting it all together

```
<form action="cgi/some.cgi" method="GET">
   <p> Please enter some text:
   <input type="text" name="string"></p>
<input type="submit">
</form>
```


Please enter some text:

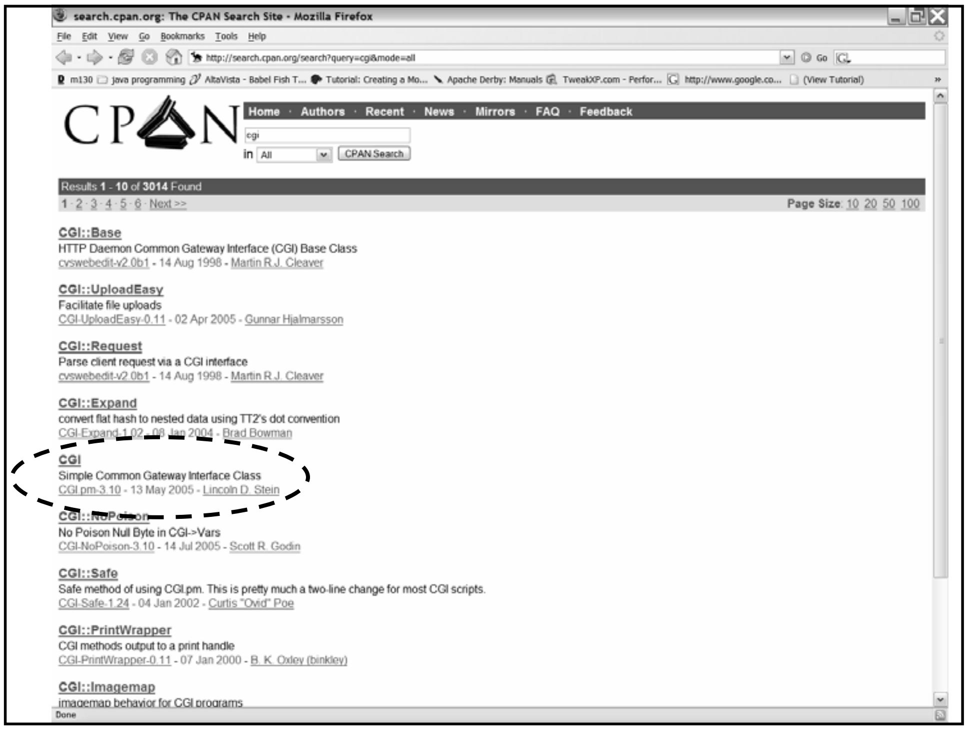
test - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
file:///C:/DOCUME~1/Shlomo/MYDOCU~1/TEACHI~1/3157-F~1/homework/te.html
Getting Started Latest Headlines dealsdujour:: Over ...
Submit Query
Done

Decoding Form Input

1. $ENV{QUERY_STRING}
2. If($ENV{REQUEST_METHOD} eq POST)
   { read $ENV{CONTENT_LENGTH}}
3. Split pairs around &
4. Split keys and values
5. Decode URL
6. Remember key,values

Drawback

- A lot of work
- Pain if we have multiple values associated with one key
- Must be easier way…..
- CGI.pm
  - Included after 5.003_07+


search.cpan.org: The CPAN Search Site - Mozilla Firefox
File Edit View Go Bookmarks Tools Help
http://search.cpan.org/search?query=cgi&mode=all
Go
m130 java programming AltaVista - Babel Fish T... Tutorial: Creating a Mo... Apache Derby: Manuals TweakXP.com - Perfor... http://www.google.co... (View Tutorial)
CPAN
Home · Authors · Recent · News · Mirrors · FAQ · Feedback
cgi
in All
CPAN Search
Done

Results **1** - **10** of **3014** Found

1 · 2 · 3 · 4 · 5 · 6 · Next >>   **Page Size**: 10 20 50 100

CGI::Base
HTTP Daemon Common Gateway Interface (CGI) Base Class
cvswebedit-v2.0b1 - 14 Aug 1998 - Martin R.J. Cleaver

CGI::UploadEasy
Facilitate file uploads
CGI-UploadEasy-0.11 - 02 Apr 2005 - Gunnar Hjalmarsson

CGI::Request
Parse client request via a CGI interface
cvswebedit-v2.0b1 - 14 Aug 1998 - Martin R.J. Cleaver

CGI::Expand
convert flat hash to nested data using TT2's dot convention
CGI-Expand-1.02 - 08 Jan 2004 - Brad Bowman

CGI
Simple Common Gateway Interface Class
CGI.pm-3.10 - 13 May 2005 - Lincoln D. Stein

CGI::NoPoison
No Poison Null Byte in CGI->Vars
CGI-NoPoison-3.10 - 14 Jul 2005 - Scott R. Godin

CGI::Safe
Safe method of using CGI.pm. This is pretty much a two-line change for most CGI scripts.
CGI-Safe-1.24 - 04 Jan 2002 - Curtis "Ovid" Poe

CGI::PrintWrapper
CGI methods output to a print handle
CGI-PrintWrapper-0.11 - 07 Jan 2000 - B. K. Oxley (binkley)

CGI::Imagemap
imagemap behavior for CGI programs

CGI.pm

- Allows you to handle cgi in a standard format
- Can save and load key,value pairs to standard file
- Helps in creating html documents to the server by streamlining certain operations and keeping it in an object oriented design

The bad news

- Can’t use it in this class
- Want you to practice doing it the manual way…better for learning and later CGI + C/CPP

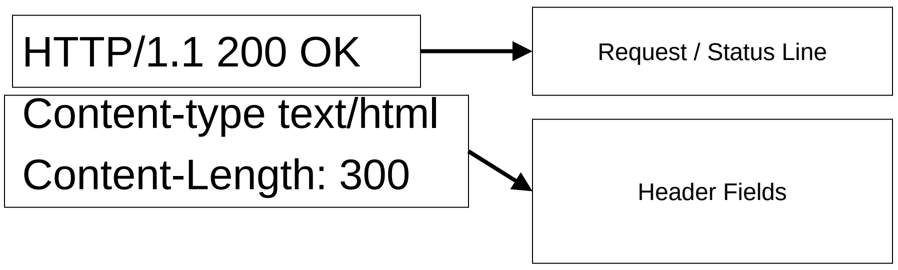
Summary: CGI

- Minimum the web server needs to provide to allow an external process to create WebPages.
- Goal: responding to queries and presenting dynamic content via HTTP.

Requirements

- Webserver setup correctly
  - Will not talk about it in class.
- Configure the cgi script
  - Will cover this lab.
- Basic http/html knowledge

http headers



GET /index.html HTTP/1.1

- GET
- HEAD
- POST
- PUT
- DELETE
- CONNECT
- OPTIONS
- TRACE

Server responses

```
HTTP/1.1 200 OK
Date: Sun, 25 Sep 2005 20:30:12 GMT
Server: Apache/1.3.5 (Unix)
Last-Modified: Wed, 20 May 1998 13:12:11 GMT
ETag: “2345-7227363ed”
Content-Length: 141
Content-Type: text/html

<HTML>
<HEAD><TITLE>…….
```

CGI Environment

- In perl available through the %ENV global hash
- Changing any of the values will only be seen by your own subprocess
  - Why?
- Some of the variables will be blank
  - Why?

File handling

- We covered basic file handling
- How does this change over the web?

File Locking

```
use Fcntl “:flock”;

open FILE, “?????.txt” or die $!;

#one of these
flock FILE, LOCK_EX;
flock FILE, LOCK_SH;
…..
flock FILE, LOCK_UN;
```

Side Note: Line Endings

- Carriage return \r
- Line Feed \n
- CRLF
- Unix – LF (\n) CR (\r)

- print “Content-type: text/html\n\n”

- Why not \n\r\n\r ????

Serving web pages

```
#!/usr/local/bin/perl
use strict;
$|=1;

my $time = localtime;
my $remote_id = $ENV{REMOTE_HOST}| $ENV{REMOTE_ADDR};
print "Content-type: text/html\n\n";

print <<END_OF_PRINTING;
This is the time : $time <P>
and your id is $remote_id

END_OF_PRINTING
```

Serving more than webpages

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



print “Content-type: image/jpeg\n\n”;
print “Content-type: image/png\n\n”;
print “Content-type: audio/mp3\n\n”;
```

Serving mp3 files

```
open(MP3FILE,”….”) || die ….

my $buffer;
print “Content-type: audio/mp3\n\n”;
binmode STDOUT;
while( read(MP3FILE, $buffer, 16384)){
 print $buffer;
}
```

Example

- http://..../cgi-bin/mp3server.cgi/Song.mp3

Argument passing

- Say you have a cool program which you can hook to the web…..
  - Give a cell phone
  - Give a message
  - Will send the cell phone a message

```
<HTML><HEAD>
<TITLE>Cool</TITLE>
</HEAD>
<BODY>

<form action="cgi-bin/cool.cgi" method="GET">
<p>Enter cell phone to use:
<input type="text" name="cellphone"></p>
<p>Enter Message:
<input type="text" name"message"></p>
<input type="submit">
</form>
</BODY></HTML>
```

```
Use CGI;
my $coolp = ‘/usr/local/bin/cellmsg’;

my $q = new CGI;
my $cell = $q->param(“cellphone”);
my $msg = $q->param(“message”);
#error checking here
open PIPE, “$coolp $cell $message |” or die “Can
   not open cellphone program”;
print $q->header( “text/plain”);
print while <PIPE>
close PIPE;
```

What can go wrong?

- When executing command can in theory pass in the following arguments

Something ; rm –rf *.*

Perl Taint mode

- -T
  - Taints all data references (incoming)
- #!/usr/bin/perl –wT
- Flags data to make sure perl doesn't do anything insecure

Tainted?

- STDIN
- CGI

- If variables/values are tainted
- Tainted follows it around with assignments

```
Sub is_tainted {
   my $var = shift;
   my $blank = substr($var ,0,0);
   return not eval { eval “1 || $blank” || 1};
}
```

Why

- Why would you want to keep track of tainted data?

Getting out of taint

- Match related patterns ($1,$2 ..)
- Idea: would check for security problems and then allow it
- Reminder: only in taint mode if set

Other issues

- Remember with each user, your perl script is being instantiated and executed
- In general might want to be able to run alongside yourself (not only in web context).
  - How do we share a variable between instances (to pass information) ?

Command shell

- A better way of executing command shell arguments to a program is to divide the work
- Create an instance of the program you want to run
- Pass arguments directly to it, instead of using the command shell (where can combine multiple commands

fork/exec

```
my $pid = open PIPE, “-|”;
die “problem forking $!” unless defined $pid;

unless($pid) {
  exec COOL, $message or die “cant open
  pipe $!”;
```

Some more background

- When you work with CGI, many times you have to work with specific formats and files
- Need to know how it will be handled on client side

- One such common file, is graphics..

Graphics

- Formats:
  - GIF (Graphic Interchange Format)
    - 256 colors
    - LZW compression
    - Animation
    - Transparent bit
  - PNG (Portable Network Graphic)
    - 256 color / 16-bit gray / 48-bit true color
    - NOT LZW
    - Alpha channels
    - Interlacing algorithms

- JPEG (Joint Photographic Expert Group)
  - 24-bit color
  - Lossy compression
  - No animation/transparency
- PDF (Portable Document Format)
  - Postscript language for document layout

Image manipulation

- Many packages in perl to work with image data
- GD
  - Lightweight package
  - Port of c graphics library
  - Manipulation routines for PNG

CGI

- CGI is a common framework
- Perl is not the only player
- We will also be doing CGI + PERL|C|CPP

Alternatives

- ASP
  - Created by Microsoft for its servers
  - Mix code into html
  - Visual basic/javascript
- PHP
  - Apache webserver
  - Similar to perl
  - Embed code in html

Alt II

- Coldfusion
  - Webserver interprets std coldfusion call embedded in html, and can add code to run custom functions
  - Windows, and linux
- Java servelts
  - Compiled java classes invoked by web client
  - Code creates documents
- FastCGI
  - Threaded instance of perl continuasly running to help cgi perl run faster
- Mod_perl
  - Appache server perl thread to make perl cgi faster

Next time

- See you in lab
  - Will be programming CGI
- Do reading please.