Introduction to Web Technologies
CGI – Common Gateway Interface

• An html web page is static (unchanging)
  – Text document sent from server to browser

• CGI program creates dynamic information
  – Program is executed upon demand
  – Generates fresh content for each request
CGI Overview

• Developer creates an HTML page with a <FORM> on it
  – Specifies the name of the program
  – Names some variables that can hold data

• User enters information into the Web page (fills in the variables in the <FORM> and clicks <SUBMIT>)

• Browser sends the information to the URL
CGI Overview continued

- Server unpacks the HTTP message
  - Finds the name of the program to call
  - Finds the data
- Server calls the program and passes in the data
- Program generates output and writes it to “standard out” (the screen, usually)
- Server takes the output and passes it along to the browser
- Browser displays the output on the user’s screen
Forms

- Forms are part of regular HTML documents
- There may be more than one form in a document
- Forms may not be nested

```html
<Form action="url"> … </Form>
```
INPUT

• Forms receive input from the user
• Each input area has its own name and type of input it may receive
• Forms may receive input from
  – Text
  – Radio
  – Checkbox
  – Submit
  – Reset
  – Password
Submitting Information via a FORM

• When SUBMIT is pushed, the contents of the form get sent to the server in the form:
  \texttt{programname?var1=value1&var2=value2}

• You may send the data via POST or GET
  – You choose this when you write the HTML page with the FORM in it
POST and GET

- **POST**
  - cgi program reads from stdin (i.e., the keyboard)
  - No limit on the amount of data sent

- **GET**
  - Cgi program reads from an environment variable (QUERY_STRING)
  - Limit on length of data sent (1,000? Characters?)

- Recommend that you use POST
A Simple Perl Program

tiny.pl

#!/usr/bin/perl

use strict; # compile time checking
use warnings; # runtime checking
my $username; # declare the variable

print "What is your username? "; # print out the question
$username = <STDIN>; # ask for the username
chomp($username); # remove "new line"
print "Hello, $username.\n"; # print out the greeting
Making It a CGI program

tinyPL.cgi

#!/usr/bin/perl

use strict;
use warnings;
use CGI qw(:standard);

my $username;

$username = "Susan";
print header();
print start_html("First CGI program");
print "Hello $username.
";
print end_html();
Calling it from a Web Page

<html>
<body bgcolor=white>

<A HREF="http://www.ittc.ku.edu/~sgauch/cgibin/cgicode/tinyPL.cgi">
Call tinyPL.cgi</a>

</body>
</html>
What the Web Page Looks Like

Call tinyPL.cgi
#!/usr/bin/perl
use strict;
use CGI qw(:standard);

my $cgiform = new CGI;
my $username = $cgiform->param("username");  #Gets values from form
my $password = $cgiform->param("password");

print header();
print start_html("Add a User");
print "<h2>Username: $username</h2>
";
print "<h2>Password: $password</h2>
";
print end_html();
Form to Call adduserPL.cgi

<html>
<body bgcolor=white>
<form action="http://www.ittc.ku.edu/~sgauch/cgi-bin/cgicode/adduserPL.cgi" method="POST">
    <center>
        <table width=70% cellpadding=5 border=1 bgcolor=#008080>
            <tr>
                <td align=center>
                    <font size=+3 style=times color=white>
                        Become a Registered Customer
                    </font>
                </td>
            </tr>
        </table>
    </center>
</form>
</body>
</html>
Form to Call adduserPL.cgi 
continued

<p><table border="0">
<tr>
<td>Choose your username: </td><td><input TYPE="text" SIZE="20" NAME="username"></td>
</tr><tr>
<td>Choose your password: </td><td><input TYPE="text" SIZE="20" NAME="password"></td>
</tr>
</table></p>
<input type="submit" value="Add Me">
</body></html>
What It Looks Like

Become a Registered Customer

Choose your username: 
Choose your password: 

Add Me
Creating CGI programs

First step

Create a program that runs from the command line
Put the program in your .public_html/cgi-bin directory
Set the variables in the program
  e.g., $username = “sgauch”;
Save the output to a file
  Perl myprog.cgi > output.html
View the file in a browser (remove top two lines first)

Second step

Have a friend login and run your program from their directory
  e.g., perl /users/myusername/.public_html/cgi-bin/myprogram.cgi > output
Creating CGI programs continued

Third step
- Design a form that calls a dummy cgi
- That cgi should just print out “hello world”
- Put the form in your .public_html directory
- View the form in the browser
- Click submit
- Check that you see “hello world”

Fourth step
- In your perl program, comment out all parts of the program
- Just print the parameters to confirm you’re getting them
- Call this program from the form

Fifth step
- Remove comments and test the REAL program
Debugging CGI programs

- Permissions problems
  - inadequate permissions
  - Test this by having someone besides yourself execute the code
  - Do and ls –l on the directory structure
    - It should be drwxr-wr-x on all directories
    - Chmod 755 my directory
    - Chmod 644 for your perl program

- Path problems
  - Test this by creating and calling a really simple cgi program

- Invalid HTML produced
  - Call this and save output to a file
  - View file in a browser
Common Problems

#!/usr/bin/perl must be the first line in the file
   Even before any other comments

Remember to call the print header() function
   It must occur before any other print statements

On some systems, the filename must end .cgi not .pl
Useful Links

- www.sergey.com/web_course/
  - JumpStart to the Web Technologies
- http://www.isoc.org/internet/history/
  - History of the Internet and Web
- http://archive.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimerAll.html
  - A thorough description of HTML tags
- http://www.cgi101.com/class/
  - A good tutorial on CGI programming