

IT350 Web and Internet Programming

Fall 2005

SlideSet #13: CGI

FLASHBACK

Things we'll learn and do

- XHTML – basics, tables, forms, frames
- Cascading Style Sheets
- JavaScript
- Dynamic HTML
- CGI

Background #1: CGI Script Basics

- Common Gateway Interface (CGI)
 - “Common”: Not specific to any operating system or language
- Output file generated at runtime:
 1. When a program executed as a CGI script, standard output is redirected to client Web server
 2. Web server then redirects output to client's browser
- CGI with C++?
 - Create a web page by sending XHTML headers and formatted text to standard output using *cout*

Background #2: Simple HTTP Transaction

- Browser sends request to www.cs.usna.edu:

```
GET / HTTP/1.0
User-Agent: Mozilla/3.0 (compatible; Opera/3.0;
Windows 95/NT4)
Accept: */*
Host: 131.122.90.78
```
- If okay, server sends back response:

```
200 OK HTTP/1.0
content-length: 4303
accept-ranges: bytes
server: Apache/2.0.54 (Unix) PHP/5.0.4
last-modified: Wed, 10 Aug 2005 13:18:07 GMT
connection: close
etag: "328f-10cf-1c8181c0"
x-pad: avoid browser bug
date: Tue, 25 Oct 2005 17:58:32 GMT
content-type: text/html

<?xml version = "1.0"?> ...
```

A Simple C++ Script Source File

```
#include <iostream>

using namespace std;

int main(){

    // Inform the browser about the contents of the file; must include a blank line
    cout << "Content-Type: text/html" << endl << endl;

    // Send the correct XHTML header information to the browser
    cout << "<?xml version='1.0'?" << endl
        << "<!DOCTYPE html PUBLIC \"-//W3C//DTD XHTML 1.1//EN\" \"<< endl
        << \"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd\">" << endl;

    // Send the <tag> formatted text to the browser using correct XHTML syntax
    cout << "<html xmlns='http://www.w3.org/1999/xhtml'" << endl;
    cout << "<head>" << endl
        << "<title>IT350 Script Greeting</title>" << endl
        << "</head>" << endl;

    cout << "<body>" << endl
        << "<h1 style='text-align:center'">Hello IT350</h1>" << endl
        << "</body>" << endl;

    cout << "</html>" << endl;

    return 0;
}
```

How can CGI respond to user?

- User submits data via a form
- Form directs results to a CGI file
- Script receives data in one of two ways:
 1. Method = "GET"
Data passed via _____
 2. Method = "POST"
Data passed via _____

Receiving input via GET

```
#include <iostream>
#include <string>
#include <cstdlib>

using namespace std;

int main(){
    xhtmlHeaders("IT350 demo: receiving info");

    string qString = getenv("QUERY_STRING");
    cout << "<h2>The form input is:</h2>" << endl;
    cout << "<p>" << qString << "</p>" << endl;

    cout << "</body>" << endl << "</html>" << endl;

    return 0;
}
```

Example

1. Enter “red monkey” into search text box
2. Yields requests for URL <http://test.com/search.cgi?q=red+monkey&lang=en>
3. CGI program output:
The form input is:

Receiving input via POST

```
#include <iostream>
#include <string>
#include <cstdlib>

using namespace std;

int main(){
    xhtmlHeaders("IT350 demo: receiving info");

    string qString = "";
    int count;

    if (getenv("CONTENT_LENGTH")){
        count = atoi(getenv("CONTENT_LENGTH"));
        for (int i=0; i<count; i++){
            qString = qString + (char) cin.get();
        }
    }

    cout << "<h2>The form input is:</h2>" << endl;
    cout << "<p>" << qString << "</p>" << endl;

    cout << "</body>" << endl << "</html>" << endl;

    return 0;
}
```

Example

1. Enter “red monkey” into search text box
2. Yields requests for URL <http://test.com/search.cgi>
3. CGI program output:
The form input is:

(empty space)

What functions would make our life easier?

1. `xhtmlHeaders(title)`

Using the helper functions

```
// Section of C++ CGI Script used to create the body of a CGI generated web
page
cout << "<body>" << endl
    << "<h1 style=\"text-align:center\">"
    << "IT350 CGI-Script Demo: Receiving Input</h1>" << endl;

cout << "<h2>The form input is:</h2>" << endl;
cout << "<p>" << qString << "</p>" << endl;
cout << "<h2>Contact Info:</h2>" << endl;
cout << "<p>" << getParam(qString,"firstname") << " "
    <<          getParam(qString,"lastname") << endl
    << "<br />" << "Email: " << getParam(qString,"email") << endl
    << "<br />" << "Sports: " << getParam(qString,"sport") << endl;

cout << "</p> </body>" << endl;
```

getQueryString()

```
string getQueryString(){
    int count;
    string qString = "";
    string method = string(getenv("REQUEST_METHOD"));

    if(method=="GET"){
        qString.append(getenv("QUERY_STRING"));
    }else{
        if(getenv("CONTENT_LENGTH"){
            count = atoi(getenv("CONTENT_LENGTH"));
            for(int i=0;i<count;i++){
                qString = qString + (char) cin.get();
            }
        }
    }
    return qString;
}
```

Pros and Cons of CGI

Parlez-vous CGI?

```
#!C:\Perl\bin\perl
# Fig. 25.15: fig25_15.pl
# Program to track the number of times
# a Web page has been accessed.

use CGI qw( :standard );

open( COUNTREAD, "counter.dat" );
$data = <COUNTREAD>;
$data++;
close( COUNTREAD );

open( COUNTWRITE, ">counter.dat" );
print( COUNTWRITE $data );
close( COUNTWRITE );

print( header(), "<div style = \"text-align: center;
                font-weight: bold\">" );
print( "You are visitor number", br() );

for ( $count = 0; $count < length( $data ); $count++ ) {
    $number = substr( $data, $count, 1 );
    print( img( { src => "images/$number.gif" } ), "\n" );
}

print( "</div>" );
```