

TEACHING BRIEF

Engaging Introductory Programming Students with CGI

T. Grandon Gill

*Information and Decision Sciences Department, College of Business Administration,
University of South Florida, 4202 East Fowler Avenue, CIS1040, Tampa, FL 33620-7800,
e-mail: ggill@coba.usf.edu*

INTRODUCTION

The technical component of Management Information Systems (MIS) programs can be challenging to design. Unlike computer science students, MIS majors tend to take relatively few technical courses, yet are expected to graduate with a broad array of computer skills. To meet these expectations, foundation courses are often designed to serve multiple objectives. For example, as a first course in an upper division undergraduate MIS major, the author teaches an introductory programming course in the C++ language. Beyond teaching the basic elements of programming and computer architecture, that course is expected to explore data representation (a foundation for data communications), the file-handling concepts (the building block of databases), and basic application construction (a core concept of systems analysis and design).

To meet these multiple objectives, programming courses need to limit the number of language features introduced. A particularly common target for simplification is input and output (I/O). The development of a graphic user interface is quite complex and text-based (console) I/O is sufficient to teach most programming concepts. Unfortunately, limiting I/O in this fashion also leads to programs that appear dated by today's standards. In consequence, students can easily develop the perception that programming is a wholly uninteresting activity, unworthy of serious consideration as a career.

To counteract such a perception, exercises need to be provided that engage students in the programming process and demonstrate its potential. In this vein, one class of exercises that we have developed allows the student to write programs that interact with a Web browser. Such exercises can be easy to construct and utilize the same console I/O that students have been taught. In addition, these exercises introduce key data communication concepts (e.g., HTML) and demonstrate client-server design. In this brief, an example of such an exercise is described.

COMMON GATEWAY INTERFACE

The technique used for the exercise is based on the common gateway interface (CGI), the simplest approach available for creating programs that respond to a

Figure 1: CGI interaction.



Web form. A typical CGI process is illustrated in Figure 1. It begins when the user presses the “Submit” button on a form in the browser. The data from that form are then submitted to the host’s Web server (sent as a long text string of name-value pairs). The Web server, in turn, repackages the data string and sends it into the CGI program. In the most common method (the POST method), the data are sent by using standard input. The CGI program then runs and sends its output—normally text generated in HTML format written to standard output—back to the server. The server then sends the Web page back to the user’s browser, where it is displayed.

Common gateway interface programs are particularly attractive, from a teaching perspective, for three reasons: (1) all their I/O is done using standard library functions, meaning the student can write them just as if they were plain console programs, (2) they are surprisingly easy to build, and (3) they illustrate how standard C++ can be relevant to practical programming problems. Furthermore, even though CGI does not represent the “state of the art” in Web design—other approaches such as Active Server Pages (ASP) and J2EE are most commonly used for commercial Web applications—they do serve to illustrate all the activities that must take place for a distributed Web application to function. Indeed, the CGI approach is the building block of more advanced Web environments in much the same way that programming languages are the building blocks of richer business environments, such as database management systems.

Unfortunately, CGI applications also have a serious drawback: they only work in the presence of a Web server. For this reason, they have traditionally not been suitable for student exercises, both as a result of the complex setup required for a server and the risk that incorrect server settings would leave the user (or institutional) system vulnerable to hackers.

ServerSim

The ServerSim tool was developed specifically to eliminate the need for a Web server when CGI applications are used as class assignments. The tool is a Windows C++ application built around a stripped down, but fully functional MS Internet Explorer browser. ServerSim differs from a regular browser in two specific ways:

- It allows the user to specify specific Web addresses (URLs) as “local.” Whenever the user navigates to one of those addresses, the call is intercepted and handled locally (an example of the settings window is presented in Figure 2, which would cause any URL at the site “www.ctoolspaper.com/cgi-bin” to be handled in the associated local folder).

Figure 2: ServerSim configuration settings.

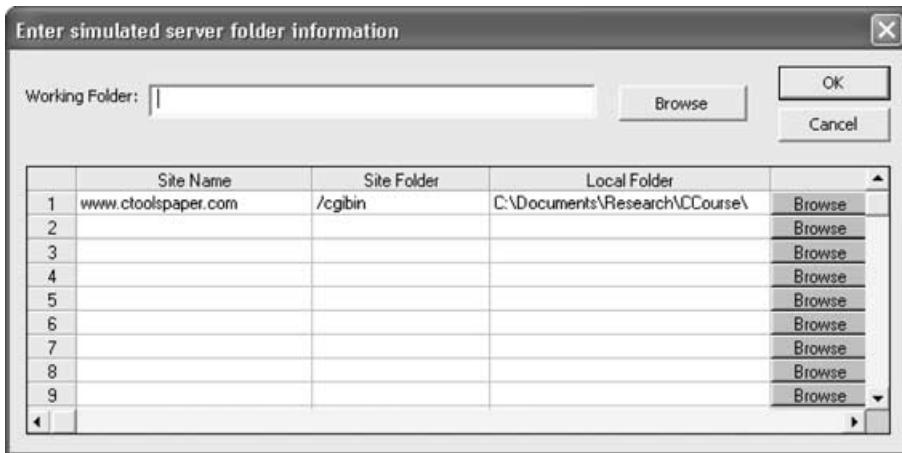


Figure 3: Form used for CGI exercise.

Do a mortgage amortization calculation

Enter the values to be calculated:

Loan amount:

Annual interest rate:

Number of years:

- When a user submits a form to one of those “local” sites, the program acts as a simulated Web server (e.g., setting environmental variables, redirecting the post string to the CGI program’s standard input, then reading the CGI program’s standard output).

Using ServerSim not only eliminates the problem of students trying to set up their own servers, it also eliminates the risk of security problems associated with novice server setups.

THE MORTGAGE AMORTIZATION EXERCISES

The CGI assignments that we use involve programming an application that generates a mortgage amortization table. Specifically, the student must write code that responds to data sent from the form shown in Figure 3 to generate the HTML page shown in Figure 4.

Figure 4: Output of student CGI program.**Summary statistics:****Loan Amount: \$300000.00****Years: 15****Interest rate: 6.000%****Monthly payment: \$2531.57****Amortization table:**

Year	Interest	Remaining Principle
1	\$17653.84	\$287274.99
2	\$16868.99	\$273765.14
3	\$16035.73	\$259422.02
4	\$15151.08	\$244194.25
5	\$14211.87	\$228027.27
6	\$13214.72	\$210863.14
7	\$12156.08	\$192640.37
8	\$11032.14	\$173293.66
9	\$9838.87	\$152753.68
10	\$8572.01	\$130946.85
11	\$7227.02	\$107795.01
12	\$5799.06	\$83215.22
13	\$4283.03	\$57119.40
14	\$2673.50	\$29414.05
15	\$964.69	\$-0.10

This page has been accessed 4 times

As preparation for the assignment, students are provided with a 75-minute Web-based video lecture that overviews the exercise. Topics covered include:

- a quick overview of HTML describing basic formatting and tables,
- an introduction to HTML forms,
- a summary of the CGI process, illustrated with a sample application that takes the variables from any form and creates a Web page containing a variable-value table, and
- an overview of ServerSim.

In addition to the lecture, students are given source code for the sample application and two multimedia files—one detailing the application they are to build and the other detailing the setup of ServerSim.

The assignment itself is broken into two parts. The first part requires the student to implement the project using C++ without classes—effectively the same as C. The second part—which is extra credit—requires recreating the same assignment using C++ classes. During Fall 2003, roughly 50% of all students completed the first part of the assignment, which was necessary in order to get an A in the course. Before any credit was given for the assignment, each student also needed to pass an oral exam with the instructor or a teaching assistant. These exams entailed the student explaining his or her code and answering questions about that code.

OUTCOMES

There are a number of indications suggesting that the CGI programming exercise is effective and well received. At the end of each semester in 2003, the instructor gathered data from students using a voluntary 300+ item instrument that students could submit for extra credit. The first part of the assignment—the only part for which a reasonable sample size is available—was rated relatively higher (62% rated it in the top half of the assignments, 30% in the top two assignments). Of greater interest were correlations found with items adapted from the Student Assessment of Learning Gains instrument (details on the instrument can be found at <http://www.wcer.wisc.edu/salgains/instructor/SALGains.asp>, last accessed on April 10, 2004). Of all seven assignments, the perception that the CGI assignment was helpful was most closely correlated with perceived gains in “confidence in your ability to work in the field” and “enthusiasm for the subject”—suggesting that students who grasped the assignment found it to be engaging.

GENERALIZABILITY

The use of CGI as a programming demonstration is not limited to C/C++. Any language capable of console I/O could be used to perform the same exercise, at least in theory. The main modification required would be a rewrite of the two simple functions that we provide to students in the chosen target language. These functions serve to: (1) set up the CGI process and (2) break up the post string into name-value pairs. The exercise would be particularly well suited for implementation in Visual Basic.NET, which now provides good support for console programming and which offers excellent built-in string manipulation functions. It could also be readily adapted to earlier text-oriented languages, such as MS-BASIC, Pascal, COBOL, or FORTRAN. ServerSim should work with any of these languages unchanged.

SAMPLE MATERIALS

ServerSim and sample source code may be obtained from the author on request, along with access to the current course Web site. The tool and instructions used in the exercises are also available as part of the Instructor’s Manual CD for the textbook “Introduction to Programming Using Visual C++.NET” by T. Grandon Gill (New York: Wiley, 2005), which includes completed projects and instructional video clips as well. [Received: April 2004. Accepted: June 2004.]