CHAPTER - 19

COMMON GATEWAY INTERFACE (CGI)

KEY OBJECTIVES:

After completing this chapter, the readers will able to—

- understand the concept of server-side programming
- get an idea about execution philosophy of CGI programs
- get an overview about languages used to write CGI programs
- understand what CGI environment variables are and their importance
- write basic CGI programs using C/C++, Perl, Python etc.
- learn how to retrieve parameters from CGI programs
- get an overview about shortcomings of CGI programs

19.1 Internet Programming paradigm

Internet programming can be classified into two categories: *client-side programming* and *server side programming*. In client-side programming paradigm, programs/scripts are downloaded, interpreted and executed by the browser. The author of programs does not have any idea about type and version of browser used to execute them. So, if the browser is not compatible with the technology used, content will not be presented properly. Let us take a specific example.

Applets are client side programming technology where special Java programs are embedded directly into web pages with the help of `<applet>` tag. When a browser loads such a web page, the

applet byte code is also downloaded and executed at the client side. If the browser uses an old JRE, applet byte-code can not be executed properly and entire thing becomes garbled. Moreover, for large applets, download time becomes significant and hence this technology tends to be unacceptable. These issues have enforced businesses to use server side programming.

19.2 Server Side Programming

Server side programming solves the above problem. The basic idea of this paradigm is that, programs are executed at the web server. Because programs are executed at the server side, there is no issue of browser incompatibility or long download time. Web server sends web pages (containing simple code generated by those programs) that even an old browser can understand.

The **C**ommon **G**ateway **I**nterface (CGI) is one of the important server-side programming techniques. The CGI connects a web server to an external application.

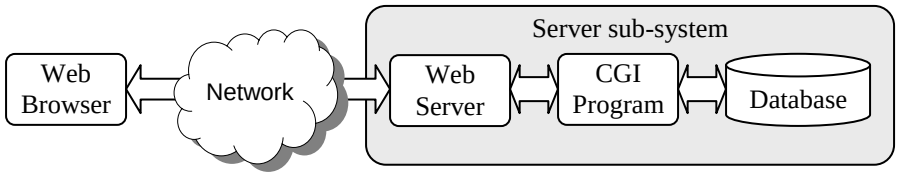

Figure 19.1: CGI Architecture

When a CGI enabled web server receives a request for a CGI program, it does not send the file as it is; instead the web server executes [Figure 19.1] the program at the server end and whatever the program outputs is sent back to the client which is then displayed in the browser's window. This simple and elegant idea can be used to develop many powerful applications.

There are numerous advantages of CGI. Most of the web servers have build in support for CGI. So, if you have your web server installed in your computer, you can start writing without any further effort.

Moreover, the CGI specification is independent of any programming language; it defines how information is transferred from web server to an external application and external application to web server. So, these external applications may be developed using a programming language that fits the application.

19.3 Languages for CGI

The most powerful feature of CGI technology is that virtually any all programming language can be used to write CGI program as long as it can read from standard input and write to standard output. Following are some popular CGI programming languages.

C/C++

C is one of the popular programming languages. It is well known for its extremely good performance. It widely used on many different software platforms and is a primary language for CGI. There is virtually no computer architecture for which a C compiler does not exists.

C++ supports Object Oriented Programming (OOP) and is suitable for large complex applications. So, if your CGI program should handle a large problem, C++ is ideal.

Perl

This interpreted language provides powerful text processing and file manipulation facilities. Perl borrows features from other programming languages such as C, shell script (sh), AWK, and sed. Due to its flexibility and adaptability, it is widely used for CGI programming. It is also used for network programming, applications that require database access, system administration and graphics programming.

Tcl

Tcl is a scripting language, originated from "Tool Command Language", but conventionally rendered as "Tcl", It is commonly used for CGI scripting as well as rapid prototyping, other scripted applications, GUIs and testing.

Python

It is an interpreted, interactive, portable, Object Oriented Programming (OOP) language. Its significant power and clear syntax makes Python an excellent instructional tool and ideal for **C**ommon **G**ateway **I**nterface (CGI) programming. Language features include modules, classes, very high level dynamic data types, and dynamic typing.

Unix/Linux Shell

A shell is a command line interpreter that provides interface to the users to execute their commands. Unix/Linux shell is extremely powerful for manipulating files, pattern seaching and matching and is ideal for CGI scripting in Unix/Linux platform.

19.3.1 Preferred Language?

The commonly used languages are Perl, C/C++, or shell script. But, your choice depends on what you want to do because different languages have different specialized features. For example, Perl is extremely powerful for string and file manipulation while C/C++ is better for complex and larger programs.

It just depends on your taste as well as facilities available on your system. If you prefer a *programming language* like C/C++ or Fortran, you have to compile the program and generate an executable code before it runs. The original source code is no longer needed. This allows you to hide your sensitive program code from others who have access to CGI directory. Moreover, the programs are already compiled, they take less time to start servicing that an interpreted one.

On the other hand, if you use any *scripting language*, such as Perl, Tcl, or Unix/Linux shell, you need to keep the script itself in the “cgi-bin” directory. Many programmers prefer CGI scripts instead of programs, as scripts are easier to modify, debug and maintain than typical compiled programs.

19.4 Applications

There are numerous applications of CGI. It is usually used to build database applications in conjunction with HTML forms. For example, suppose we want to "hook up" our database to the World Wide Web (WWW), so that people from all over the world can query it. All we need to do is to write a CGI program which can access database. The web server will execute this program to store information to the database and receive result from the database, which is sent back to the client.

19.5 Server Environment

Most of the web servers standardize the CGI mechanism. Usually a directory “cgi-bin” exists under the web server’s installation directory. The files in this directory is treated differently. Any

file requested from this special cgi-bin directory is not simply read and sent; instead it is executed in the computer where the web server is installed. The output of this program is actually sent to the browser requested this file. The program is usually is pure executable file of typically a Perl or Python script.

Though cgi-bin directory is usually considered as the container of CGI scripts, most web servers allow us to specify other directories as the CGI directory.

Suppose that you have entered the following URL in the address bar of your browser:

```
http://192.168.1.2/cgi-bin/hello.pl
```

The web server recognizes the file `hello.pl` which is nothing but a perl script in the directory `cgi-bin` directory. It then executes `hello.pl` and sends the output of this script to the browser which then displays it on the screen.

Though CGI programs are typically written on Unix platform, you can also write your CGI programs on Windows environment. For this purpose, you need a web server. Perl is the typical language used for CGI programs, you can use C/C++ or python as well. This section, we shall discuss how to install and configure Apache and Perl on your computer running Windows. One the Apache and Perl is successfully installed, you can test you CGI programs using the address `http://locahost` in your own computer. This is quite useful to test your CGI programs locally before installing them in the actual server.

Download installer file for Apache from `http://httpd.apache.org/download.cgi`. Now install it by double clicking on the installer file. We are assuming that you have installed Apache in the directory "D:\Apache Software Foundation".

Now download and install ActivePerl from http://www.activestate.com/Products/ActivePerl/. We are assuming that you have successfully installed Perl in the "D:\Perl" directory.

19.5.1 Configuring Apache

To enable CGI programs, you need to modify Apache configuration file `httpd.conf` which can be found in the `conf` directory under the apache installation directory. Go the `<Directory>…</Directory>` section. Uncomment or add the following line:

```
Options MultiViews Indexes SymLinksIfOwnerMatch Includes ExecCGI
```

The `Options` specifies what options are available in this directory. The `ExecCGI` enables the CGI scripting. Go the `AddHandler` section and add or uncomment the following line.

```
AddHandler cgi-script .cgi .pl
```

This causes files with extension `.cgi` and `.pl` to be treated as CGI programs. Finally save the configuration file and restart the Apache. Ensure that the server restarted successfully by checking `http://localhost/` in your browser.

If everything goes well, your web server is ready to serve CGI request. Save your CGI programs in the `cgi-bin` directory under the Apache installation directory.

Use the following line as the first line of your Perl program:

```
#!D:/perl/bin/perl.exe
```

This tell the web server where to find the Perl interpreter to interpret Perl programs.

19.6 Environment Variables

In order to pass and retrieve parameters, web server uses several environment variables. The web server usually sets these environment variables before starting a CGI program. CGI program can inspect those environment variables [Table 19.1] to retrieve information. Note that CGI environment variables are primary source of information which server-side program can use.

Table 19.1: CGI Environment Variables

Variable Name	Description
SERVER_NAME	The server's DNS name or IP Address
SERVER_SOFTWARE	The name and version of the web server software answering the request.
GATEWAY_INTERFACE	The version of the CGI specification that the server complies.
SERVER_PROTOCOL	The HTTP version used by the server
SERVER_PORT	The port number used by the web server
CONTENT_TYPE	The type of the data of the content. It is used when the client is sending attached content to the server e.g example file upload etc.
CONTENT_LENGTH	The length of the query information is case of POST method
HTTP_ACCEPT	The list of MIME types that the client can accept
HTTP_USER_AGENT	The browser name and version that the user is using to make this request
PATH	The value of PATH environment variable
QUERY_STRING	The URL-encoded information which follows the ? in the URL
REMOTE_ADDR	The IP address of the remote host that made the request.
REMOTE_PORT	The port number from which the request was sent
REQUEST_METHOD	The HTTP method used in the request. The most common methods are GET, HEAD and POST.
SCRIPT_FILENAME	The full path of the CGI script being executed.
SCRIPT_NAME	The relative path of the script being executed

Following Perl script prints available environment variables with their values.

```
#!D:/perl/bin/perl.exe
print "Content-type: text/html\n\n";
print "<table style=\"font-size:12; font-family:arial\" border=\"0\" >";
print "<caption>CGI variables</caption>";
foreach $var (sort(keys(%ENV))) {
    $val = $ENV{$var};
    print "<tr><td>${var}</td><td >${val}</td></tr>\n";
}
print "</table>";
```

Don't think about the syntax of the above Perl program. It simply iterates through the associative array variable ENV that contains all the environment variables and prints their values as a tabular format. We shall discuss the syntax briefly later in this chapter. The above script results the output as shown in the figure 19.2

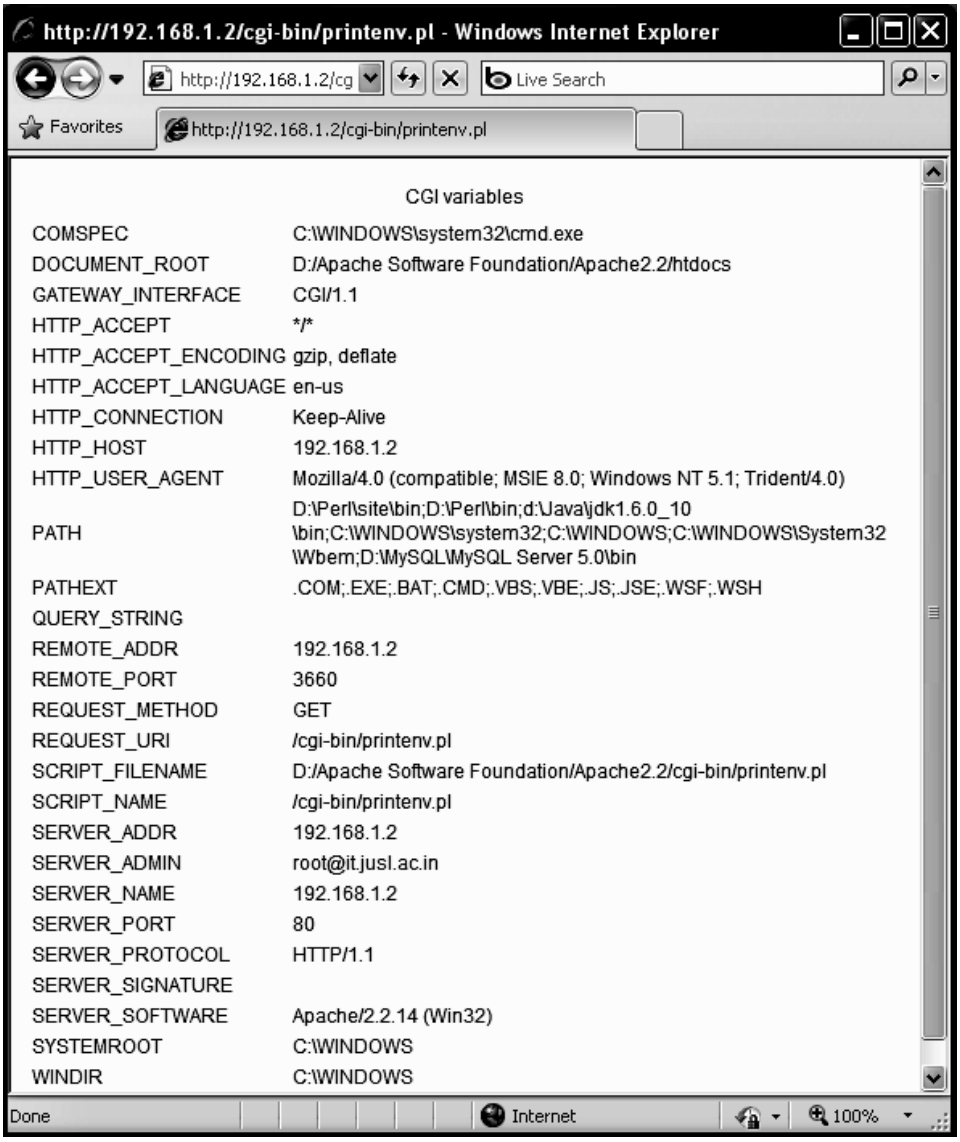

Figure 19.2: CGI Environment variables

19.7 CGI building blocks

Any CGI script consists of basically three steps:

- Read parameters passed to this script
- Process these parameters
- Write HTML response to the standard output.

19.8 CGI Scripting using C, shell script

Writing CGI programs using C/C++ language is somehow different than writing the same in shell script. Here shell means Unix/Linux shell. Windows shell is not so much powerful and is hardly used. Note that shell scripts are interpreted by the underlying shell. You simply need to mention which shell should be used to interpret your script as the first line of your script as follows:

```
#!/bin/bash
```

C/C++ programs are, on the other hand, compiled programs. Writing CGI programs using C/C++ consists of basically two steps:

- Compile the program to generate executable file
- Put this executable file in the cgi directory.

Compiling a C/C++ program is different for different compiler. Ask your system administrator to know the compilation procedure of C/C++ programs. Moreover, C/C++ programs are not platform independent. So, you must compile you program in the computer where the web server runs using a suitable compiler available in that machine.

19.9 Writing CGI programs

In this section, we shall develop simple programs to display "Hello World!" using different languages such as Perl, C and Python. Following is the program (`hello.pl`) written in Perl.

```
#!D:/perl/bin/perl.exe
print "Content-type: text/html\n\n";
print "<html>\n";
print "  <head><title>First Perl script</title></head>\n";
print "  <body>\n";
print "     <h2>Hello, World!</h2>\n";
print "  </body>\n";
print "</html>\n";
```

The `hello.pl` is a simple script that prints a simple HTML document on the standard output i.e. screen. The first line of any Perl script should look like this:

```
#!D:/perl/bin/perl.exe
```

The `#!` indicates that this is a script. The `D:/perl/bin/perl.exe` refers to the full path name of the Perl interpreter to be used by the web server to execute the Perl script. But this could be different in different systems. Unix or Linux OS, this is typically `/usr/bin/perl` or `/usr/local/bin/perl`. If you are not sure about the path, type "whereis perl" or "which perl" at the command prompt, it shows the path where the Perl interpreter is stored. Alternatively, ask the webmaster regarding the location of the Perl interpreter.

The rest part consists of actual perl statements. Before printing anything else, you should use the following line:

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

This prints `Content-type: text/html` followed by a blank line. This sent as a HTTP response header which specifies the type of the content to be displayed in the browser's screen. In our case, we shall send an HTML document and that is why the `Content-type` header is specified as `text/html`. The above Perl program, except Content-type header, basically generates the following:

```
<html>
   <head><title>First Perl script</title></head>
```

```
   <body>
      <h2>Hello, World!</h2>
   </body>
</html>
```

If you enter the following URL, you will see the result as shown in the figure 19.3

```
http://192.168.1.2/cgi-bin/hello.pl
```

where 192.168.1.2 is the IP address of the computer where the web server is running. If you are testing your program locally, you can also use the following URL

```
http://localhost/cgi-bin/hello.pl
```

or alternatively

```
http://127.0.0.1/cgi-bin/hello.pl
```

Figure 19.3: Hello World CGI program using Perl

Let us now write a CGI program (hello.c) using C language for the same purpose.

```
int main() {
  printf("Content-type: text/html\n\n");
  printf("<html>\n");
  printf("  <head><title>First Perl script</title></head>\n");
  printf("  <body>\n");
  printf("     <h2>Hello, World!</h2>\n");
  printf("  </body>\n");
  printf("</html>\n");
  return 0;
}
```

Compile the program using suitable compiler such as Microsoft's Visual C++. Microsoft's Visual C++ compiler will generate the executable file hello.exe. Rename this file to hello and put it in the CGI directory. If you are using Unix/Linux platform, use the following command to compile the C program and generate an executable file.

```
gcc –o hello hello.c
```

It generates an executable file hello. Put this file in the CGI directory. Now, type the following URL, you will see the same output as shown in the figure 19.4.

```
http://192.168.1.2/cgi-bin/hello
```

Figure 19.4: Hello World CGI program using C

The following program is written in Python (`hello.cgi`) to generate the same output.

```
#!D:\Python31\python.exe
print("Content-type: text/html\n\n");
print("<html>");
print("  <head><title>Python demo<title></title></head>");
print("  <body>");
print("     Hello World!");
print("  </body>");
print("</html>");
```

The first line indicates the location of the Python interpreter. We are assuming that you have installed the Python in the "D:\Python31" directory. Put this file in the CGI directory and use the following URL to view result.

```
http://192.168.1.2/cgi-bin/hello.cgi
```

19.9.2 Getting arguments

In this section, we shall discuss how CGI programs can retrieve URL encoded parameters passed to them. The parameters passed have the following characteristics:

- They are sent in the form of name and value pair.
- Each name/value pair starts with an & sign
- The name and value are separated by the equal (=) sign

URL encoding changes some special characters to placeholders and replaces to hexadecimal values. To retrieve the information, CGI program should implement the following basic steps:

- Get the information from proper environment variable depending upon the method type. For example, we can retrieve information from the `QUERY_STRING` environment variable for GET method.
- Change all placeholders to their correct values.
- Split each group of name/value pair.
- Convert hexadecimal values back to their original characters.
- Find the respective name and value.

Following C program (`add.c`) shows how to retrieve two parameter values. The program adds those two values and sends the result back to the client

```
#include <stdio.h>
#include <stdlib.h>
int main(void) {
   long a, b;
```

```
   printf("Content-Type:text/html\n\n");
   char *data = getenv("QUERY_STRING");
   sscanf(data,"a=%d&b=%d", &a, &b);
   printf("%d + %d = %ld", a, b, a + b);
   return 0;
}
```

Here we first obtain the entire parameter information using `getenv()` function. The individual parameter is then obtained using `sscanf()` function. Compiler the program to have an executable file add using the procedure described before. Now, use the following URL to test the program.

```
http://192.168.1.2/cgi-bin/add?a=3&b=4
```

The result is shown in the figure 19.5.

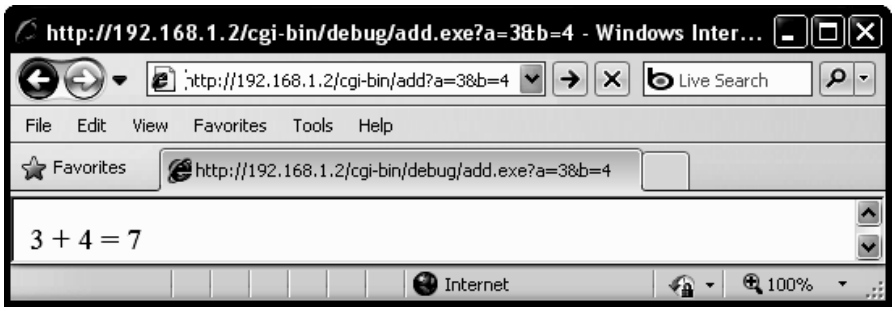

Figure 19.5: Retrieving parameters

Following is the equivalent Perl program.

```
#!D:/perl/bin/perl.exe
use CGI qw(:standard);
print "Content-type: text/html\n\n";
$a = param('a');
$b = param('b');
$c = $a + $b;
print "$a + $b = $c";
```

Following Perl script retrieves all parameters appended to the URL.

```
#!D:/perl/bin/perl.exe
print "Content-type:text/html\r\n\r\n";
$buffer = $ENV{'QUERY_STRING'};
@pairs = split(/&/, $buffer);
foreach $pair (@pairs)    {
   ($name, $value) = split(/=/, $pair);
   print "$name=$value<br>";
}
```

19.10 CGI Security

CGI programs are executables, which allow the world run programs on your system. This is not safest things to do at all. Therefore, some security precautions need to be taken when we talk about CGI programs. Note that CGI programs are kept in a special directory, so that the web server can execute the program rather than just sending them to the browser. This is the most important concept the CGI writers and external users try to make use to do evil things. The CGI directory is usually controlled by the webmaster who should prohibit the average user from

creating and running CGI programs. There are many ways to allow access to CGI scripts, but webmaster decides how to set these up for you.

Although CGI protocol is inherently secure, CGI scripts may become major source of security holes. CGI scripts should be written with utmost care just as the server. Web administrator should not also trust on the script writers and should not install arbitrary CGI scripts.

If you are a system administrator, webmaster or are otherwise involved with the administration of a network, you should create a security policy for your web site. The following points should be taken in consideration while writing the security policy:

- Users who can use the system
- When they are allowed to use the system
- What things they are allowed to do
- A way to grant access to the system
- Acceptable permission to use the file system
- Procedure for monitoring system
- Actions to be takes to suspected security breaches

19.11 Alternatives And Enhancements to CGI

Each time the web server receives a CGI request, it starts a new process that serves the request. Starting up a new process may take significant amount of time and memory than actual task of generating output. If users send CGI request very frequently, they may quickly overwhelm the web server

If the CGI program is a interpreted program such as Perl, python or shell script, it needs more time. We can use compiled programs such as C/C++ instead of scripting languages to avoid overhead involved due to interpretation.

The overhead due to process creation may be reduced by using technology such as FastCGI or by using special extension modules provided by some web server.

FastCGI uses a single persistent process that handles many request during its life time. This way overhead due to a new process creation for every request may be avoided. Multiple simultaneous requests can be handled either by using multiple processes or using a single process with internal multiplexing. FastCGI allows web server to perform simple operations such as reading a file before the request is handed over the FastCGI program. Some of the web servers that implement FastCGI are Apache HTTP Server, Microsoft IIS, Resin Application Server, Sun Java System Web Server etc.

Another alternative to CGI is Simple Common Gateway Interface (SCGI) which is similar to FastCGI but easier to implement. In this technology, clients sends request to the SCGI server using SCGI request message. The server sends the response back and closes the connection. Web servers that implement SCGI are Apache HTTP Server, Lighttpd, Cherokee etc.

Another viable and effective solution to CGI is Java servlets. Servlets can avoid overhead due to new process creation. Web server loads the servlet class when it starts up. The servlet can then serve many requests using a separate thread. Since threads are lightweight than processes, servlets are more efficient than CGI programs.

Java Server Page (JSP) is an extension to Java servlet and is a potential solution to CGI technology. In the next two chapters, we shall discuss Java servlet and JSP separately.

19.12 Key Words

Common Gateway Interface—A standard which interfaces the HTTP server software with CGI programs which run on the server.

Client-side programming—A programming paradigm where programs are executed at the client machine

Server-side programming—A programming paradigm where programs are executed at the server machine

Gateway program—A program that the web server contacts to process some request sent by the client

CGI Languages—The programming languages that can be used to write CGI programs.

CGI directory—CGI programs are kept in some predefined special directory so that web server knows that those programs must be executed instead of sending then directly to the clients

Compiled CGI program—A program which is compiled to generate executable code that acts as a CGI programs.

Interpreted CGI program—A program whose statements are interpreted using an interpreter program

CGI Environment variables—The variables used to pass and retrieve information in CGI programs

SCGI—A extension to CGI called Simple Common Gateway Interface

CGI alternatives—The competent technologies such as servlets, JSP, ASP, PHP etc.

19.13 Summary

The Common Gateway Interface (CGI), which is one of the popular server-side technologies. There are numerous advantages of CGI. Most of the web servers have build in support for CGI. Moreover, the CGI specification is independent of any programming language; it defines how information is transferred from web server to an external application and external application to web server. So, these external applications may be developed using a programming language that fits the application.

When a CGI enabled web server receives a request for a CGI program, it does not send the file as it is; instead the web server executes the program and whatever the program outputs is sent back to the client which is then displayed in the browser's window.

The most powerful feature of CGI technology is that virtually any all programming language can be used to write CGI program as long as it can read from standard input and write to standard output. Some of the examples are C/C++, Perl, Python, Tcl, shell, Ruby etc. Compiled programs such as C/C++ run faster than interpreted programs. Interpreted programs, on the other hand, are easier to modify, debug and maintain.

Most of the web servers standardize the CGI mechanism. Usually a directory "cgi-bin" exists under the web server's installation directory. The files in this directory is treated differently. Any file requested from this special cgi-bin directory is not simply read and sent; instead it is executed in the computer where the web server is installed. The output of this program is actually sent to the

browser requested this file. The program is usually is pure executable file of typically a Perl or Python script.

In order to pass and retrieve parameters, web server uses several environment variables. The web server usually sets these environment variables before starting a CGI program. CGI program can inspect those environment variables to retrieve information.

Since CGI program can be written in almost all languages and in all platforms, exact procedure of executing CGI program varies in platform as well as web server.

CGI programs are executables, which allow the world run programs on your system. Since, CGI programs are executables, some security precautions need to be taken when we talk about CGI programs.

19.14 Web Resources

`http://www.ietf.org/rfc/rfc3875.txt`
The Common Gateway Interface (CGI) Version 1.1

`http://hoohoo.ncsa.illinois.edu/cgi/`
The Common Gateway Interface

`http://www.w3.org/CGI/`
CGI: Common Gateway Interface

`http://www.fastcgi.com/devkit/doc/fcgi-spec.html`
FastCGI Specification

19.15 Exercises

19.15.1 Objective Type Questions

1. Server-side scripting is about "programming" the behavior of the :
   a) Server
   b) Browser
   c) HTML
   d) All of the above

2. What is the full form of CGI?
   a) Common Graphical Interface
   b) Cascading Gateway Interface
   c) Class Generator Instance
   d) Common Gateway Interface

3. Which of the following defines how a Web server communicates with external applications?
   a) HTTP
   b) CGI
   c) SMTP
   d) TCP/IP

4. Which of the following is true regarding CGI?
   a) It is a client-side technology

b) It is a name of a browser
c) It is a server-side technology
d) It is a web server

5. Which of the following languages can be used as CGI language?
   a) C
   b) C++
   c) Perl
   d) All of the above

6. Which of the following directories is usually used to store CGI programs?
   a) cgi-bin
   b) cgi
   c) cgiprogs
   d) bin

7. Which of the following environment variables is used to retrieve URL encoded information?
   a) QUERY
   b) QUERY_STRING
   c) PARAM
   d) URL_ENCODE

8. What should be the Content-type to generate HTML documents?
   a) text/html
   b) text/plain
   c) plain/text
   d) plain/html

9. Whenever web server receives a CGI request, it
   a) Creates a new thread
   b) Uses an existing process
   c) Uses an existing thread
   d) Creates a new process

10. Which of the following statements is true regarding CGI?
    a) Compiled program is faster than interpreted program
    b) Compiled program is slower than interpreted program
    c) The source code is necessary for compiled program
    d) The source code is not necessary for interpreted program

11. Which of the following is a compiled language?
    a) C++
    b) Perl
    c) Python
    d) Shell

12. Which of the following is an interpreted language?
    a) Tcl
    b) Ruby
    c) Visual Basic

d) All of the above

13. Which of the following header should be set by a CGI script?
    a) Value-type
    b) Content
    c) Content-type
    d) Return-type

14. Which of the following CGI environment variable represent the port number used by the web server?
    a) REMOTE_PORT
    b) SERVER_PORT
    c) PORT
    d) HOST_PORT

15. Which of the following function in C/C++ is used to retrieve the value of an environment variable?
    a) env()
    b) environment()
    c) getenv()
    d) getvariable()

16. Which of the following lines should be printed by a CGI program to generate an HTML document?
    a) Content-type: html/text
    b) Content-type: text/html
    c) Return-type: text/html
    d) Return-type: html/text

17. The first line of any CGI script should start with
    a) $!
    b) &!
    c) @!
    d) #!

18. Which of the following characters is used to separate parameters from URL?
    a) ?
    b) &
    c) $
    d) %

19. Which of the following characters is used to separate parameter name and its value?
    a) ^
    b) $
    c) =
    d) &

20. Which of the following functions is used in Perl to print a string on the standard output?
    a) print
    b) show

c) display
d) out

19.15.2 Subjective Type Questions

1. How does server-side programming differ from client-side programming?
2. What are the relative advantages and disadvantages of compiled and interpreted programs in CGI?
3. Mention two names of each compiled and interpreted languages.
4. What are Environment variables? Name two CGI environment variables.
5. What is the purpose of Content-type header in CGI programs?
6. Can we use Java as the CGI language? If yes, how?
7. Mention two web servers that support CGI programming.
8. Describe how parameters are passed and retrieved in CGI programs.
9. What are the drawbacks of CGI technology? How can we overcome them?
10. Mention some of the alternatives of CGI technology.