Perl and CGI (Common Gateway Interface)

Outline

Objectives

- you will learn:
  - To understand basic Perl programming.
  - To understand the Common Gateway Interface.
  - To understand string processing and regular expressions in Perl.
  - To be able to read and write cookies.
  - To be able to construct programs that interact with MySQL databases.

Introduction

- Practical Extraction and Report Language (Perl)
  - One of the most widely used language for Web programming
- Common Gateway Interface (CGI)
  - Standard interface through which users interact with applications on Web servers
  - Provides way for clients to interact with applications on Web server
  - CGI script
    - Can be written in many different languages, including Perl

Introduction

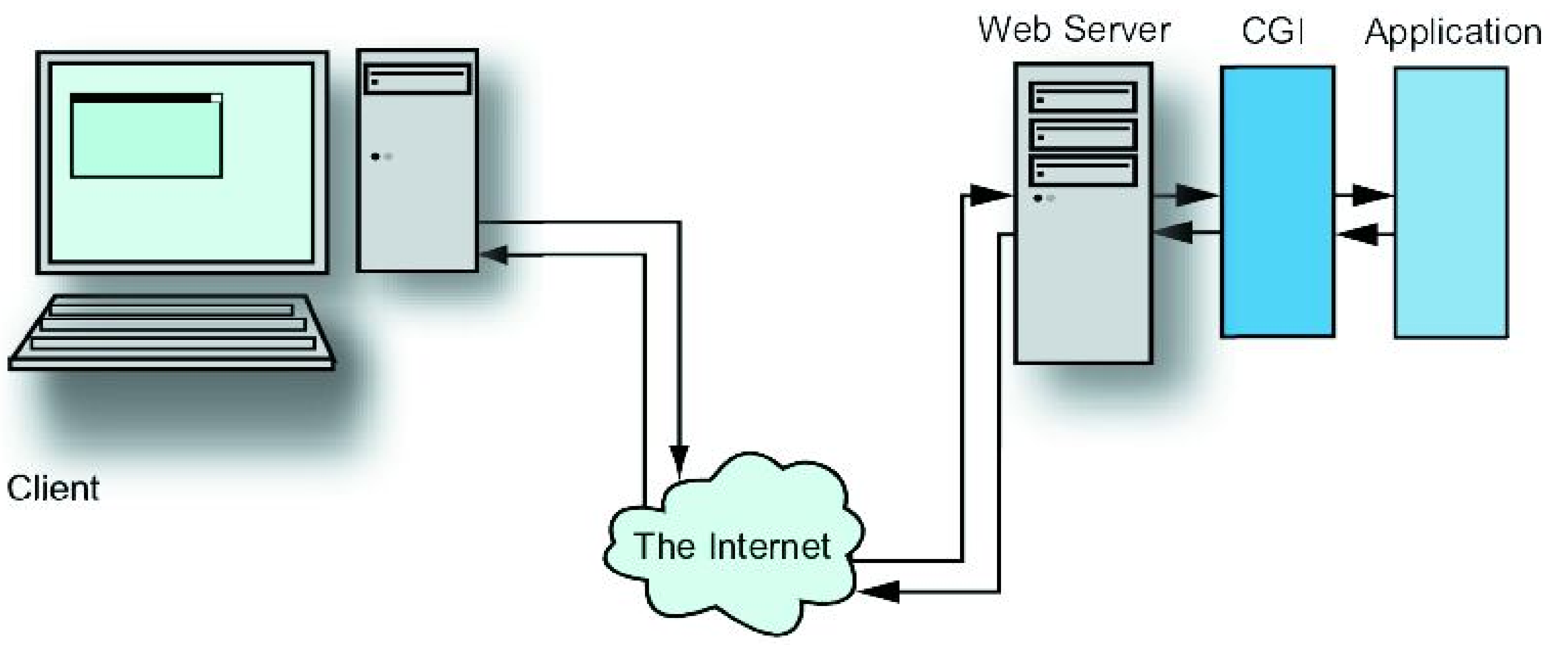


Fig. 25.1 Data path of a typical CGI-based application.

Perl

- Case sensitive
- Comment character (#)
  - Instruct interpreter to ignore everything on current line following #
  - Allows programmers to write descriptive comments in programs
  - “shebang” construct (`#!`)
    - Indicates the path to the Perl interpreter
- `print`
  - Write text to screen
- Escape sequence `\n`
  - Moves output cursor to next line

Perl

- Interpolation
  - Replace variable with its associated data
- `undef`
  - In numeric context
    - Evaluates to 0
  - In a string context
    - Empty string (“”)
- Range operator (`..`)
  - Specifies all values between uppercase A and uppercase Z are to replace in array

fig25_02.pl
(1 of 1)

```
[illegible]

# A first program in Perl.

print( "Welcome to Perl!\n" );
```

```
Welcome to Perl!
```

Perl

Data type	Format for variable names of this type	Description
Scalar	$*scalarname*	Can be a string, an integer number, a floating-point number or a reference.
Array	@*arrayname*	An ordered list of scalar variables that can be accessed using integer indices.
Hash	%*hashname*	An unordered set of scalar variables whose values are accessed using unique scalar values (i.e., strings) called **keys**.

Fig. 25.3 Perl data types.

```perl
#!C:\Perl\bin\perl
# Fig. 25.4: fig25_04.pl
# Program to illustrate the use of scalar variables.

$number = 5;
print( "The value of variable \$number is: $number\n\n" );

$number += 5;
print( "Variable \$number after adding 5 is: $number\n" );

$number *= 2;
print( "Variable \$number after multiplying by 2 is: " );
print( "$number\n\n\n" );

# using an uninitialized variable in the context of a string
print( "Using a variable before initializing: $variable\n\n" );

# using an uninitialized variable in a numeric context
$test = $undefined + 5;
print( "Adding uninitialized variable \$undefined " );
print( "to 5 yields: $test\n" );

# using strings in numeric contexts
$string = "A string value";
$number += $string;
```

fig25_04.pl
(1 of 2)

```perl
print( "Adding a string to an integer yields: $number\n" );

$string2 = "15charactersand1";
$number2 = $number + $string2;
print( "Adding $number to string \"$string2\" yields: " );
print( "$number2\n" );
```

fig25_04.pl
(2 of 2)

```
The value of variable $number is: 5

Variable $number after adding 5 is: 10
Variable $number after multiplying by 2 is: 20


Using a variable before initializing:

Adding uninitialized variable $undefined to 5 yields: 5
Adding a string to an integer yields: 20
Adding 20 to string "15charactersand1" yields: 35
```

```perl
#!C:\Perl\bin\perl
# Fig. 25.5: fig25_05.pl
# Program to demonstrate arrays in Perl.

@array = ( "Bill", "Bobby", "Sue", "Michelle" );

print( "The array contains: @array\n" );
print( "Printing array outside of quotes: ", @array, "\n\n" );

print( "Third element: $array[ 2 ]\n" );

$number = 3;
print( "Fourth element: $array[ $number ]\n\n" );

@array2 = ( 'A' .. 'Z' );
print( "The range operator is used to create a list of\n" );
print( "all capital letters from A to Z:\n" );
print( "@array2 \n\n" );

$array3[ 3 ] = "4th";
print( "Array with just one element initialized: @array3 \n\n" );

print( 'Printing literal using single quotes: ' );
print( '@array and \n', "\n" );
```

fig25_05.pl
(1 of 2)

```perl

print( "Printing literal using backslashes: " );
print( "\@array and \\n\n" );
```

```
The array contains: Bill Bobby Sue Michelle
Printing array outside of quotes: BillBobbySueMichelle

Third element: Sue
Fourth element: Michelle

The range operator is used to create a list of
all capital letters from A to Z:
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

Array with just one element initialized:    4th

Printing literal using single quotes: @array and \n
Printing literal using backslashes: @array and \n
```

fig25_05.pl
(2 of 2)

String Processing and Regular Expressions

- Text manipulation
  - Done with a regular expression
    - Series of characters that serves as a pattern-matching template
- String-processing tasks
  - Can be accomplished by using Perl’s equality and comparison operators
- `foreach` statement
  - Iterates sequentially through elements
- Match operator (`m//`)
  - Uses regular expressions to search string for specified pattern
- Binding operator
  - Binds whatever is on its left side to a regular-expression operator on its right side

String Processing and Regular Expressions

- Metacharacters
  - Specify patterns or contexts that cannot be defined using literal characters
- Word boundary
  - Boundary between an alphanumeric character and something that is not an alphanumeric character
- + modifier
  - Quantifier that instructs Perl to match preceding character one or more times

```perl
#!C:\Perl\bin\perl
# Fig. 25.6: fig25_06.pl
# Program to demonstrate the eq, ne, lt, gt operators.

@fruits = qw( apple orange banana );

foreach $item ( @fruits ) {

   if ( $item eq "banana" ) {
      print( "String '$item' matches string 'banana'\n" );
   }

   if ( $item ne "banana" ) {
      print( "String '$item' does not match string 'banana'\n" );
   }

   if ( $item lt "banana" ) {
      print( "String '$item' is less than string 'banana'\n" );
   }

   if ( $item gt "banana" ) {
      print( "String '$item' is greater than string 'banana'\n" );
   }
}
```

```
String 'apple' does not match string 'banana'
String 'apple' is less than string 'banana'
String 'orange' does not match string 'banana'
String 'orange' is greater than string 'banana'
String 'banana' matches string 'banana'
```

```perl
#!C:\Perl\bin\perl
# Fig 25.7: fig25_07.pl
# Searches using the matching operator and regular expressions.

$search = "Now is is the time";
print( "Test string is: '$search'\n\n" );

if ( $search =~ /Now/ ) {
   print( "String 'Now' was found.\n" );
}

if ( $search =~ /^Now/ ) {
   print( "String 'Now' was found at the beginning of the line." );
   print( "\n" );
}

if ( $search =~ /Now$/ ) {
   print( "String 'Now' was found at the end of the line.\n" );
}

if ( $search =~ /\b ( \w+ ow ) \b/x ) {
   print( "Word found ending in 'ow': $1 \n" );
}

```

fig25_07.pl
(1 of 2)

```perl
if ( $search =~ /\b ( \w+ ) \s ( \1 ) \b/x ) {
   print( "Repeated words found: $1 $2\n" );
}

@matches = ( $search =~ / \b ( t \w+ ) \b /gx );
print( "Words beginning with 't' found: @matches\n" );
```

```
Test string is: 'Now is is the time'

String 'Now' was found.
String 'Now' was found at the beginning of the line.
Word found ending in 'ow': Now
Repeated words found: is is
Words beginning with 't' found: the time
```

fig25_07.pl
(2 of 2)

String Processing and Regular Expressions

Quantifier	Matches
`{n}`	Exactly `n` times
`{m,n}`	Between `m` and `n` times inclusive
`{n,}`	`n` or more times
`+`	One or more times (same as `{1,}`)
`*`	Zero or more times (same as `{0,}`)
`?`	One or zero times (same as `{0,1}`)

Fig. 25.8 Some of Perl's quantifiers.

Symbol	Matches	Symbol	Matches
`^`	Beginning of line	`\d`	Digit (i.e., `0` to `9`)
`$`	End of line	`\D`	Nondigit
`\b`	Word boundary	`\s`	Whitespace
`\B`	Nonword boundary	`\S`	Nonwhitespace
`\w`	Word (alphanumeric) character	`\n`	Newline
`\W`	Nonword character	`\t`	Tab

Fig. 25.9 Some of Perl's metacharacters.

String Processing and Regular Expressions

Modifying character	Purpose
`g`	Performs a global search; finds and returns all matches, not just the first one found.
`i`	Ignores the case of the search string (case insensitive).
`m`	The string is evaluated as if it had multiple lines of text (i.e., newline characters are not ignored).
`s`	Ignores the newline character and treats it as whitespace. The text is seen as a single line.
`x`	All whitespace characters are ignored when searching the string.

Fig. 25.10 Some of Perl's modifying characters.

Viewing Client/Server Environment Variables

- Environment variables
  - Contain information about execution environment in which a script is being run
- `use` statement
  - Instructs Perl programs to include modules
  - Modules
    - Contents of predefined packages
  - `import tag :standard`
    - Import a predefined set of standard functions
  - Key
    - Value name
    - Assigned a value using the arrow operator (`=>`)

Viewing Client/Server Environment Variables

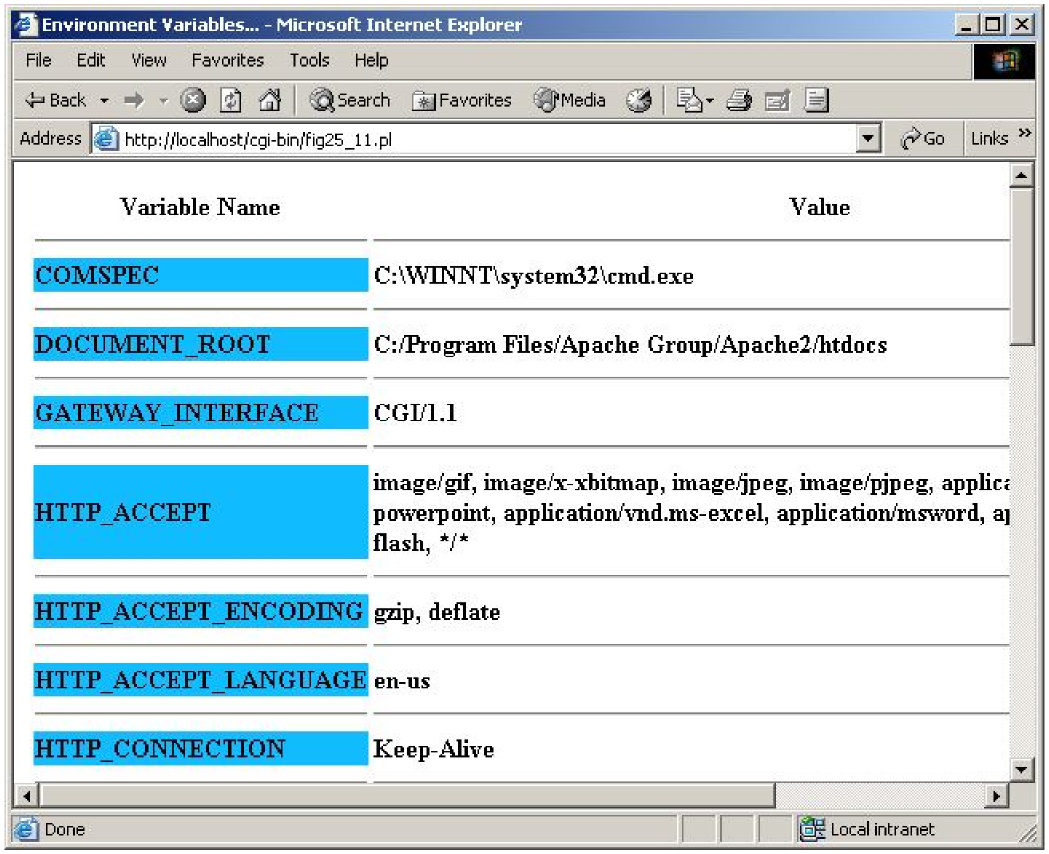
- `%ENV` hash
  - Built-in table in Perl that contains names and values of all environment variables
- Function `sort`
  - Order array of keys alphabetically

```perl
#!C:\Perl\bin\perl
# Fig. 25.11: fig25_11.pl
# Program to display CGI environment variables.

use CGI qw( :standard );

$dtd =
"-//W3C//DTD XHTML 1.0 Transitional//EN\"
   \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

print( header() );

print( start_html( { dtd => $dtd,
                    title => "Environment Variables..." } ) );

print( "<table style = \"border: 0; padding: 2;
        font-weight: bold\">" );

print( Tr( th( "Variable Name" ),
           th( "Value" ) ) );

print( Tr( td( hr() ), td( hr() ) ) );

foreach $variable ( sort( keys( %ENV ) ) ) {

```

fig25_11.pl
(1 of 2)

```
   print( Tr( td( { style => "background-color: #11bbff" },
                    $variable ),
              td( { style => "font-size: 12pt" },
                       $ENV{ $variable } ) ) );

   print( Tr( td( hr() ), td( hr() ) ) );
}

print( "</table>" );
print( end_html() );
```

fig25_11.pl
(2 of 2)



Variable Name	Value
COMSPEC	C:\WINNT\system32\cmd.exe
DOCUMENT_ROOT	C:/Program Files/Apache Group/Apache2/htdocs
GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, applica powerpoint, application/vnd.ms-excel, application/msword, ap flash, */*
HTTP_ACCEPT_ENCODING	gzip, deflate
HTTP_ACCEPT_LANGUAGE	en-us
HTTP_CONNECTION	Keep-Alive

Environment Variables... - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/cgi-bin/fig25_11.pl
Go Links
Done
Local intranet

Form Processing and Business Logic

- XHTML forms
  - Enable Web pages collect data from users and send to Web server for processing by server-side programs and scripts
  - Function `param`
    - Part of Perl CGI module
    - Retrieves values from a form field’s value
  - Business logic (business rules)
    - Design of verifying information
  - Function `br`
    - Adds a break (`<br />`) to XHTML page
  - Functions span and div
    - Adds `<span>` and `<div>` to page respectively

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<!-- Fig. 25.12: fig25_12.html -->

<html>
   <head>
      <title>Sample form to take user input in XHTML</title>
   </head>

    <body style = "font-face: arial; font-size: 12pt">

       <div style = "font-size: 14pt; font-weight: bold">
             This is a sample registration form.
       </div>

       <br />
       Please fill in all fields and click Register.

       <form method = "post" action = "/cgi-bin/fig25_13.pl">

          <img src = "images/user.gif" /><br />

          <div style = "color: blue" >
             Please fill out the fields below.<br />
```

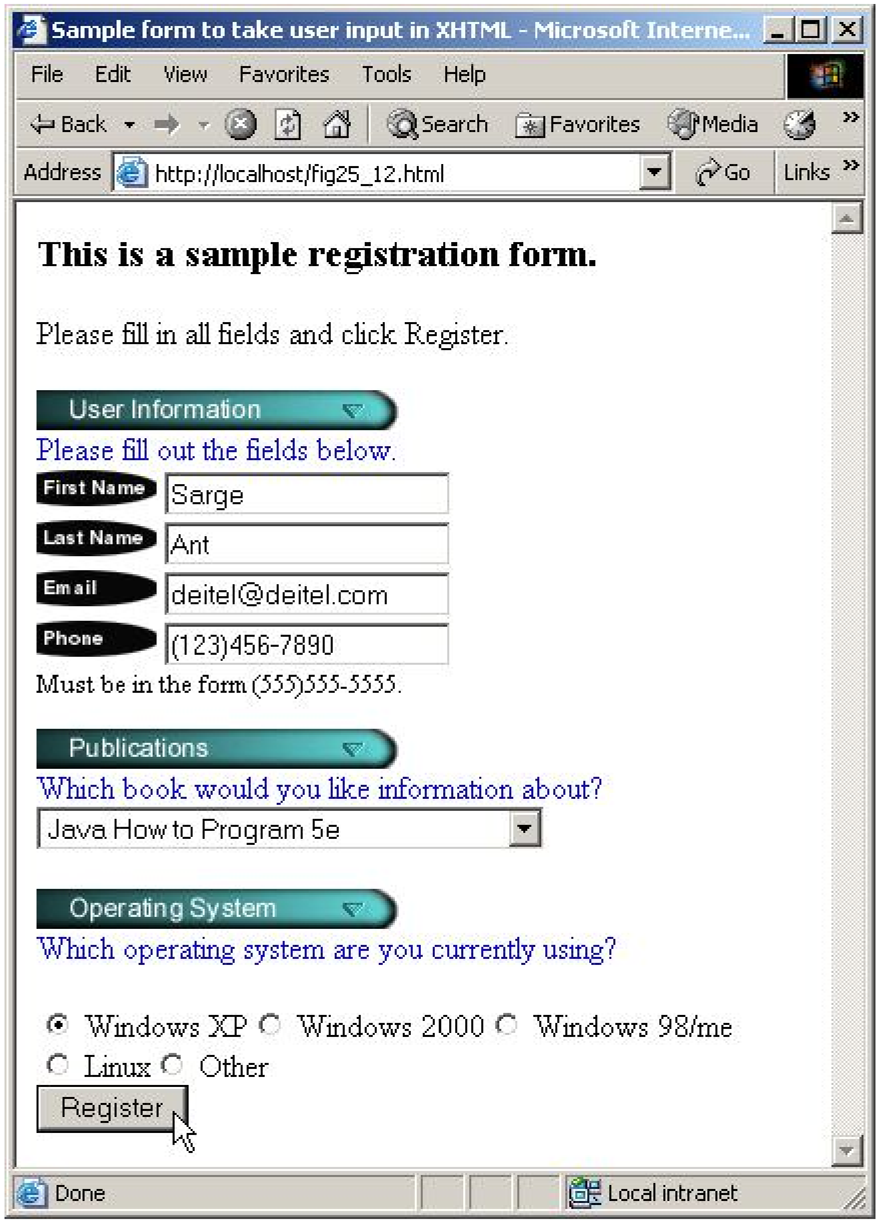
fig25_12.html
(1 of 3)

```
         </div>

         <img src = "images/fname.gif" />
         <input type = "text" name = "fname" /><br />
         <img src = "images/lname.gif" />
         <input type = "text" name = "lname" /><br />
         <img src = "images/email.gif" />
         <input type = "text" name = "email" /><br />
         <img src = "images/phone.gif" />
         <input type = "text" name = "phone" /><br />

         <div style = "font-size: 10pt">
            Must be in the form (555)555-5555.<br /><br />
         </div>

         <img src = "images/downloads.gif" /><br />
         <div style = "color: blue">
            Which book would you like information about?<br />
         </div>

         <select name = "book">
            <option>Internet and WWW How to Program 3e</option>
            <option>C++ How to Program 4e</option>
            <option>Java How to Program 5e</option>
            <option>XML How to Program 1e</option>
```

fig25_12.html
(2 of 3)

```
         </select><br /><br />

         <img src = "images/os.gif" /><br />
         <div style = "color: blue">
            Which operating system are you currently using?
         </div><br />

         <input type = "radio" name = "os"
            value = "Windows XP" checked />
         Windows XP<input type = "radio"
            name = "os" value = "Windows 2000" />
         Windows 2000<input type = "radio"
            name = "os" value = "Windows 98/me" />
         Windows 98/me<br /><input type = "radio"
            name = "os" value = "Linux" />
         Linux<input type = "radio" name = "os"
            value = "Other" />
         Other<br /><input type = "submit"
            value = "Register" />
      </form>
   </body>
</html>
```

fig25_12.html
(3 of 3)


Sample form to take user input in XHTML - Microsoft Interne...
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/fig25_12.html
Go Links
Done
Local intranet

This is a sample registration form.

Please fill in all fields and click Register.

User Information

Please fill out the fields below.

First Name: Sarge

Last Name: Ant

Email: deitel@deitel.com

Phone: (123)456-7890

Must be in the form *(555)555-5555*.

Publications

Which book would you like information about?

Java How to Program 5e

Operating System

Which operating system are you currently using?

[✓ Windows XP] Windows 2000 Windows 98/me Linux Other

Register

```perl
#!C:\Perl\bin\perl
# Fig. 25.13: fig25_13.pl
# Program to read information sent to the server
# from the form in the fig25_12.html document.

use CGI qw( :standard );

$os = param( "os" );
$firstName = param( "fname" );
$lastName = param( "lname" );
$email = param( "email" );
$phone = param( "phone" );
$book = param( "book" );

$dtd =
"-//W3C//DTD XHTML 1.0 Transitional//EN\"
   \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

print( header() );

print( start_html( { dtd => $dtd,
                     title => "Form Results" } ) );

if ( $phone =~ / ^ \( \d{3} \) \d{3} - \d{4} $ /x ) {
   print( "Hi " );
```

fig25_13.pl
(1 of 4)

```
   print( span( { style => "color: blue; font-weight: bold" },
                  $firstName ) );
   print( "!" );

   print( "\nThank you for completing the survey." );
   print( br(), "You have been added to the " );

   print( span( { style => "color: blue; font-weight: bold" },
                  $book ) );
   print( " mailing list.", br(), br() );

   print( span( { style => "font-weight: bold" },
                 "The following information has
                  been saved in our database: " ), br() );

   print( table(
          Tr( th( { style => "background-color: #ee82ee" },
                    "Name" ),
              th( { style => "background-color: #9370db" },
                    "E-mail" ),
              th( { style => "background-color: #4169e1" },
                    "Phone" ),
              th( { style => "background-color: #40e0d0" },
                    "OS" ) ),

```

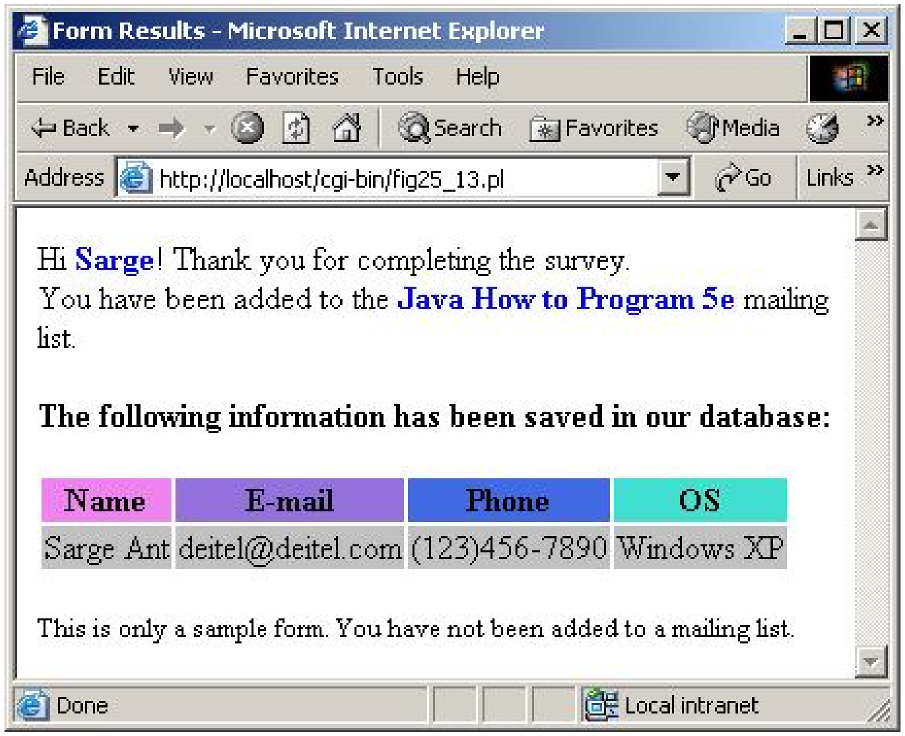
fig25_13.pl
(2 of 4)

```
          Tr( { style => "background-color: #c0c0c0" },
              td( "$firstName $lastName" ),
              td( $email ),
              td( $phone ),
              td( $os ) ) ) );

   print( br() );

   print( div( { style => "font-size: x-small" },
            "This is only a sample form. You have not been
             added to a mailing list." ) );
}
else {
   print( div( { style => "color: red; font-size: x-large" },
             "INVALID PHONE NUMBER" ), br() );

   print( "A valid phone number must be in the form " );
   print( span( { style => "font-weight: bold" },
                  "(555)555-5555." ) );

   print( div( { style => "color: blue" },
             "Click the Back button, and enter a
              valid phone number and resubmit." ) );
   print( br(), br() );
   print( "Thank you." );
```

fig25_13.pl
(3 of 4)

```
}

print( end_html() );
```

fig25_13.pl
(4 of 4)

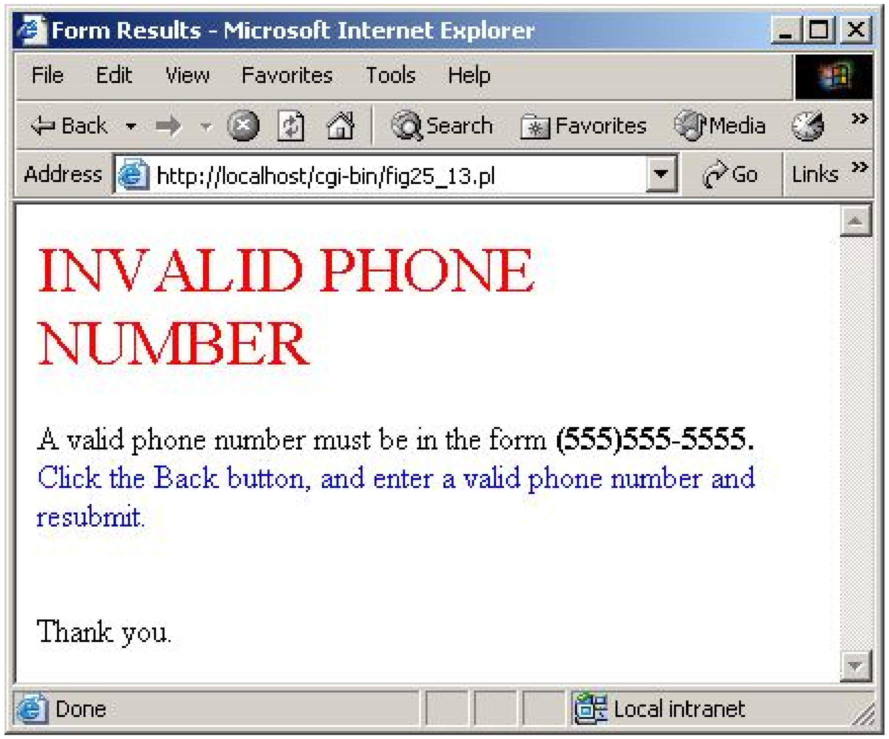

Form Results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/cgi-bin/fig25_13.pl
Go Links
Done
Local intranet

Hi Sarge! Thank you for completing the survey.
You have been added to the **Java How to Program 5e** mailing list.

The following information has been saved in our database:

Name	E-mail	Phone	OS
Sarge Ant	deitel@deitel.com	(123)456-7890	Windows XP

This is only a sample form. You have not been added to a mailing list.


Form Results - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/cgi-bin/fig25_13.pl
Go Links
Done
Local intranet

INVALID PHONE NUMBER

A valid phone number must be in the form **(555)555-5555.**
Click the Back button, and enter a valid phone number and resubmit.

Thank you.

Server-Side Includes

- Commands embedded in XHTML documents to allow creation of simple dynamic content
- Written as XHTML comments
- `.shtml` file extension (`s` stands for server)
  - Parsed by server
- `ECHO` command
  - Display variable information
- Keyword `VAR`
  - Specifies name of the variable
- `EXEC`
  - Can be used to run CGI scripts and embed their output directly into Web page

Server-Side Includes

- Diamond operator `<>`
  - Read one line of file referred to by filehandle `COUNTREAD`
- `>` character
  - Write mode
- Append mode (`>>`)
  - Appending to the end of a file
- Function `close`
  - Terminates connection
- `for` structure
  - Iterates
- Function `length`
  - Returns length of character string

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig. 25.14: fig25_14.shtml -->

<html>
   <head>
      <title>Using Server-Side Includes</title>
   </head>

    <body>
       <h3 style = "text-align: center">
          Using Server-Side Includes
       </h3>

       <!--#EXEC CGI="/cgi-bin/fig25_15.pl" --><br />

       The Greenwich Mean Time is
       <span style = "color: blue">
          <!--#ECHO VAR="DATE_GMT" -->.
       </span><br />

```

fig25_14.shtml
(1 of 3)

```
      The name of this document is
      <span style = "color: blue">
         <!--#ECHO VAR="DOCUMENT_NAME" -->.
      </span><br />

      The local date is
      <span style = "color: blue">
         <!--#ECHO VAR="DATE_LOCAL" -->.
      </span><br />

      This document was last modified on
      <span style = "color: blue">
         <!--#ECHO VAR="LAST_MODIFIED" -->.
      </span><br />

      Your current IP Address is
      <span style = "color: blue">
         <!--#ECHO VAR="REMOTE_ADDR" -->.
      </span><br />

      My server name is
      <span style = "color: blue">
         <!--#ECHO VAR="SERVER_NAME" -->.
      </span><br />

```

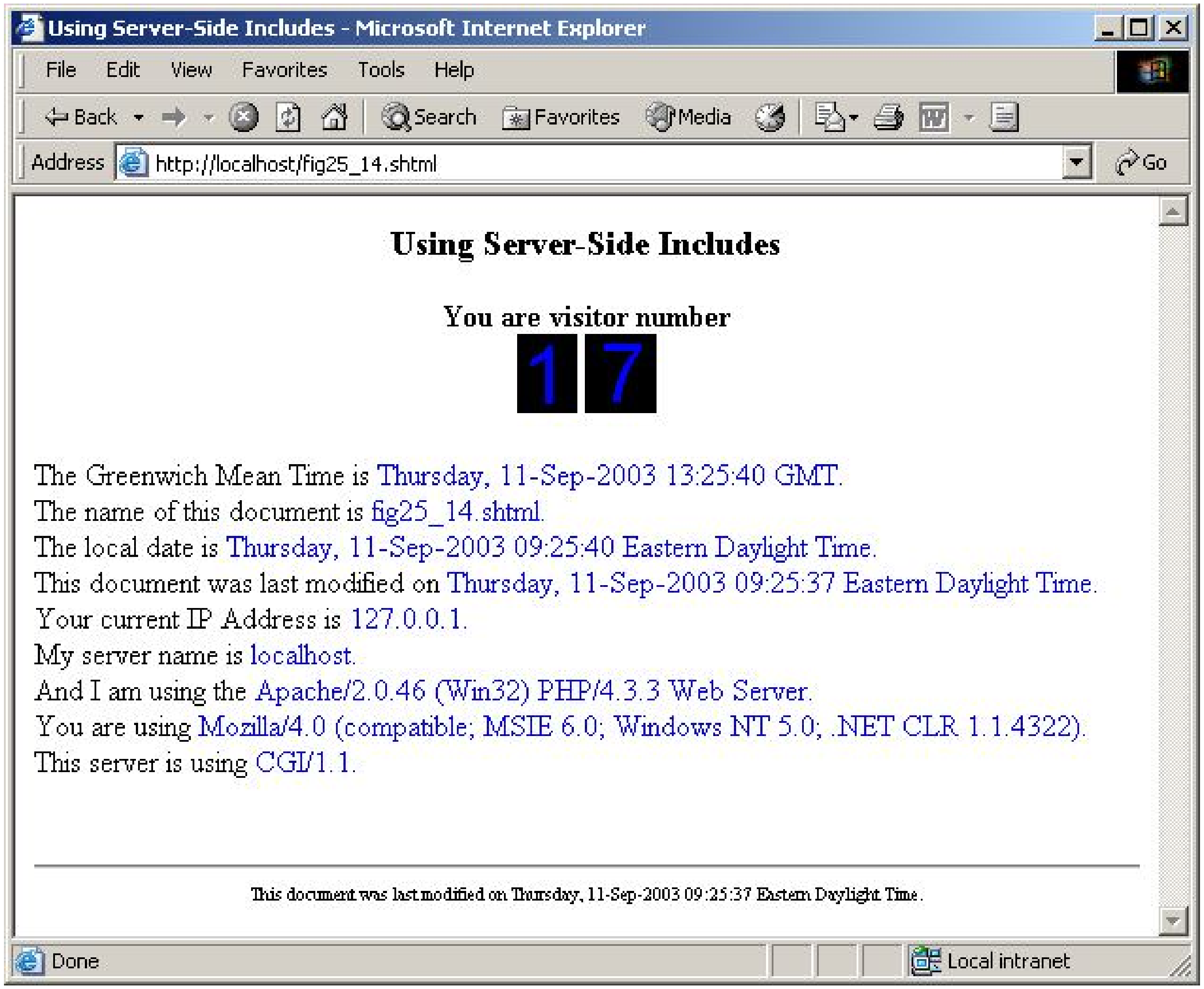
fig25_14.shtml
(2 of 3)

```
      And I am using the
      <span style = "color: blue">
         <!--#ECHO VAR="SERVER_SOFTWARE" -->
         Web Server.
      </span><br />

      You are using
      <span style = "color: blue">
         <!--#ECHO VAR="HTTP_USER_AGENT" -->.
      </span><br />

      This server is using
      <span style = "color: blue">
         <!--#ECHO VAR="GATEWAY_INTERFACE" -->.
      </span><br />

      <br /><br />
      <div style = "text-align: center;
                    font-size: xx-small">
         <hr />
         This document was last modified on
         <!--#ECHO VAR="LAST_MODIFIED" -->.
      </div>
   </body>
</html>
```

fig25_14.shtml
(3 of 3)

```
#!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>" );
```

fig25_15.pl
(1 of 1)


Using Server-Side Includes - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/fig25_14.shtml
Go
Done
Local intranet

Using Server-Side Includes

You are visitor number

17

```
The Greenwich Mean Time is Thursday, 11-Sep-2003 13:25:40 GMT.
The name of this document is fig25_14.shtml.
The local date is Thursday, 11-Sep-2003 09:25:40 Eastern Daylight Time.
This document was last modified on Thursday, 11-Sep-2003 09:25:37 Eastern Daylight Time.
Your current IP Address is 127.0.0.1.
My server name is localhost.
And I am using the Apache/2.0.46 (Win32) PHP/4.3.3 Web Server.
You are using Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.0; .NET CLR 1.1.4322).
This server is using CGI/1.1.
```

This document was last modified on Thursday, 11-Sep-2003 09:25:37 Eastern Daylight Time.

Verifying a Username and Password

- Private Web sites
  - Visible only to certain people
  - Username and password verification
    - `chomp`
      - Remove newline character at end of line
    - `split`
      - Divide string into substrings at specified separator

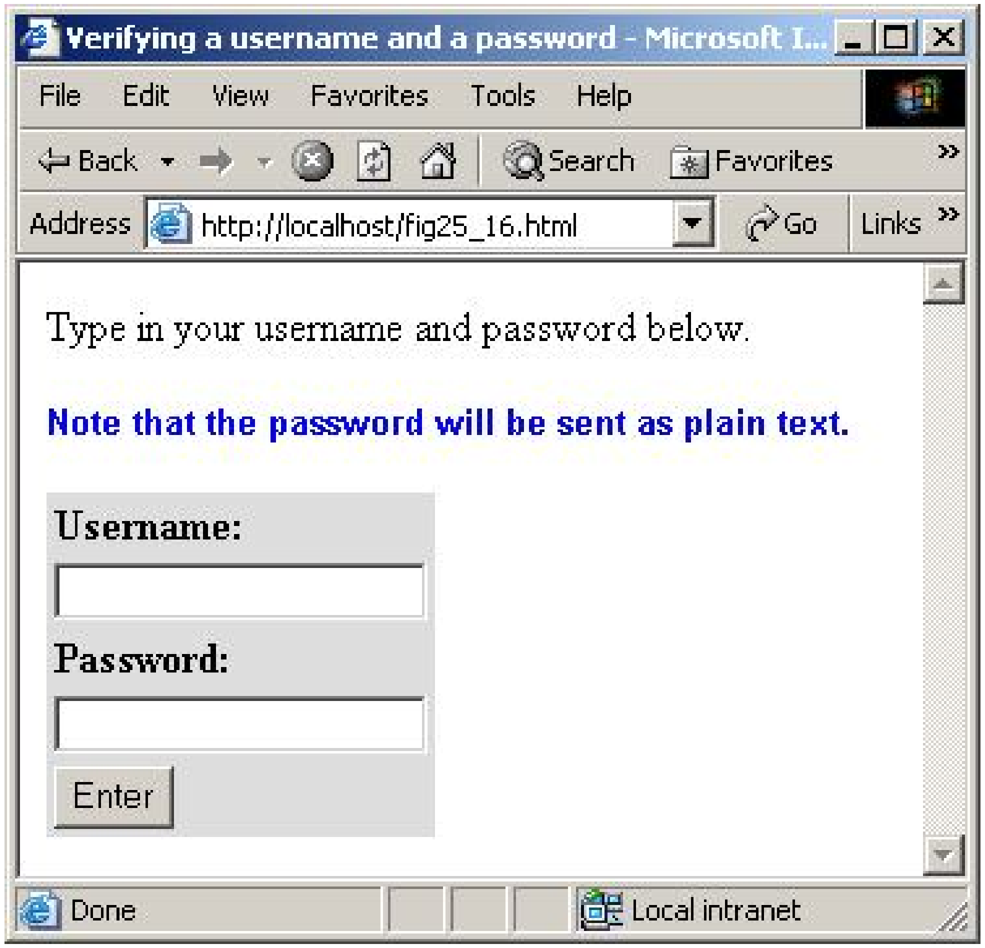
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig. 25.16: fig25_16.html -->

<html>
   <head>
      <title>Verifying a username and a password</title>
   </head>

    <body>
       <p>
          <div style = "font-family = arial">
             Type in your username and password below.
          </div><br />

          <div style = "color: #0000ff; font-family: arial;
                       font-weight: bold; font-size: small">
             Note that the password will be sent as plain text.
          </div>
       </p>

       <form action = "/cgi-bin/fig25_17.pl" method = "post">

```

fig25_16.html
(1 of 3)

```
            <table style = "background-color: #dddddd">
               <tr>
                  <td style = "font-face: arial;
                               font-weight: bold">Username:</td>
               </tr>
               <tr>
                  <td>
                     <input name = "username" />
                  </td>
               </tr>
               <tr>
                  <td style = "font-face: arial;
                               font-weight: bold">Password:</td>
               </tr>
               <tr>
                  <td>
                     <input name = "password" type = "password" />
                  </td>
               </tr>
               <tr>
                  <td>
                     <input type = "submit" value = "Enter" />
                  </td>
               </tr>
            </table>
```

fig25_16.html
(2 of 3)

```
        </form>
    </body>
</html>
```


Verifying a username and a password - Microsoft I...
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/fig25_16.html Go Links
Done
Local intranet

Type in your username and password below.

Note that the password will be sent as plain text.

Username:

Password:

fig25_16.html
(3 of 3)

```perl
#!C:\Perl\bin\perl
# Fig. 25.17: fig25_17.pl
# Program to search a database for usernames and passwords.

use CGI qw( :standard );

$dtd =
"-//W3C//DTD XHTML 1.0 Transitional//EN\"
   \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

$testUsername = param( "username" );
$testPassword = param( "password" );

open( FILE, "password.txt" ) ||
   die( "The database could not be opened." );

while ( $line = <FILE> ) {
   chomp( $line );
   ( $username, $password ) = split( ",", $line );

   if ( $testUsername eq $username ) {
      $userVerified = 1;

      if ( $testPassword eq $password ) {
         $passwordVerified = 1;
```

fig25_17.pl
(1 of 3)

```perl
            last;
         }
      }
   }

   close( FILE );

   print( header() );
   print( start_html( { dtd => $dtd,
                        title => "Password Analyzed" } ) );

   if ( $userVerified && $passwordVerified ) {
      accessGranted();
   }
   elsif ( $userVerified && !$passwordVerified ) {
      wrongPassword();
   }
   else {
      accessDenied();
   }

   print( end_html() );

   sub accessGranted
   {
```

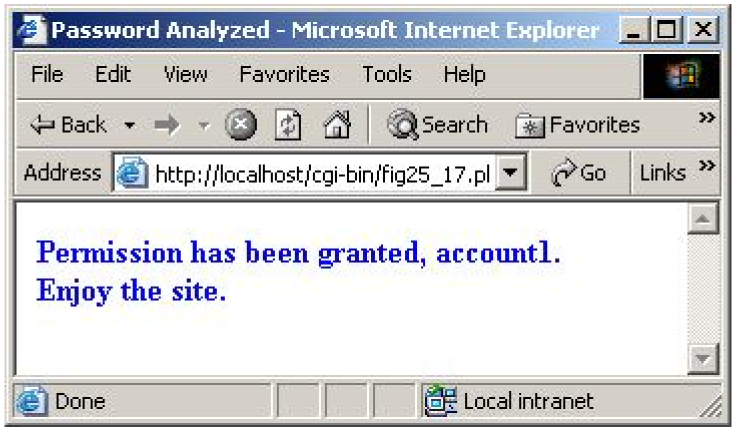
fig25_17.pl
(2 of 3)

```
   print( div( { style => "font-face: arial;
                           color: blue;
                           font-weight: bold" },
         "Permission has been granted,
         $username.", br(), "Enjoy the site." ) );
}

sub wrongPassword
{
   print( div( { style => "font-face: arial;
                           color: red;
                           font-weight: bold" },
        "You entered an invalid password.", br(),
        "Access has been denied." ) );
}

sub accessDenied
{
   print( div( { style => "font-face: arial;
                           color: red;
                           font-size: larger;
                           font-weight: bold" },
        "You have been denied access to this site." ) );
}
```

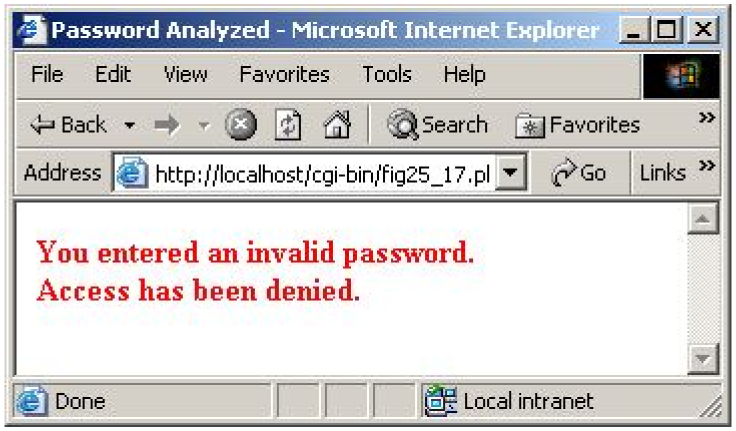
fig25_17.pl
(3 of 3)

```
account1,password1
account2,password2
account3,password3
account4,password4
account5,password5
account6,password6
account7,password7
account8,password8
account9,password9
account10,password10
```

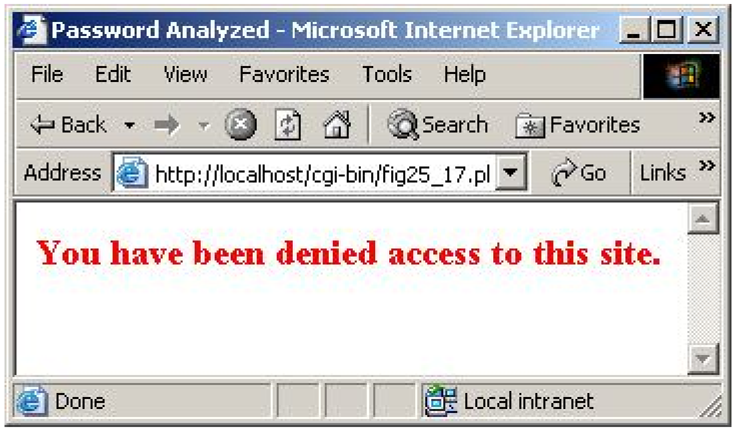
password.txt
(1 of 1)


Password Analyzed - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cgi-bin/fig25_17.pl Go Links
Done Local intranet

Permission has been granted, account1.
Enjoy the site.


Password Analyzed - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cgi-bin/fig25_17.pl Go Links
Done Local intranet

You entered an invalid password.
Access has been denied.


Password Analyzed - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cgi-bin/fig25_17.pl Go Links
Done Local intranet

You have been denied access to this site.

Using DBI to Connect to a Database

- Perl DBI (Database Interface)
    - `prepare`
    - `execute`
    - `disconnect`
    - `finish`

Using DBI to Connect to a Database

- Perl DBI (Database Interface), cont.
  - Access different types of databases uniformly
  - Perl Package Manager (PPM)
    - Download and install Perl modules and packages
    - Start PPM
      - Type `ppm` at command prompt
      - Interactive mode
    - Database handles
      - Create and manipulate connection to database
    - Statement handles
      - Create and manipulate SQL statements to database
    - DBI method `connect`
      - Connect to database

```
#!C:\Perl\bin\perl
# Fig. 25.19: fig25_19.pl
# CGI program that generates a list of authors.

use CGI qw( :standard );
use DBI;
use DBD::mysql;

$dtd =
"-//W3C//DTD XHTML 1.0 Transitional//EN\"
   \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

print( header() );

print( start_html( { dtd => $dtd,
                     title => "Authors" } ) );

# connect to "books" database, no password needed
$databaseHandle = DBI->connect( "DBI:mysql:books",
                     "root", "", { RaiseError => 1 } );

# retrieve the names and IDs of all authors
$query = "SELECT FirstName, LastName, AuthorID
          FROM Authors ORDER BY LastName";

```

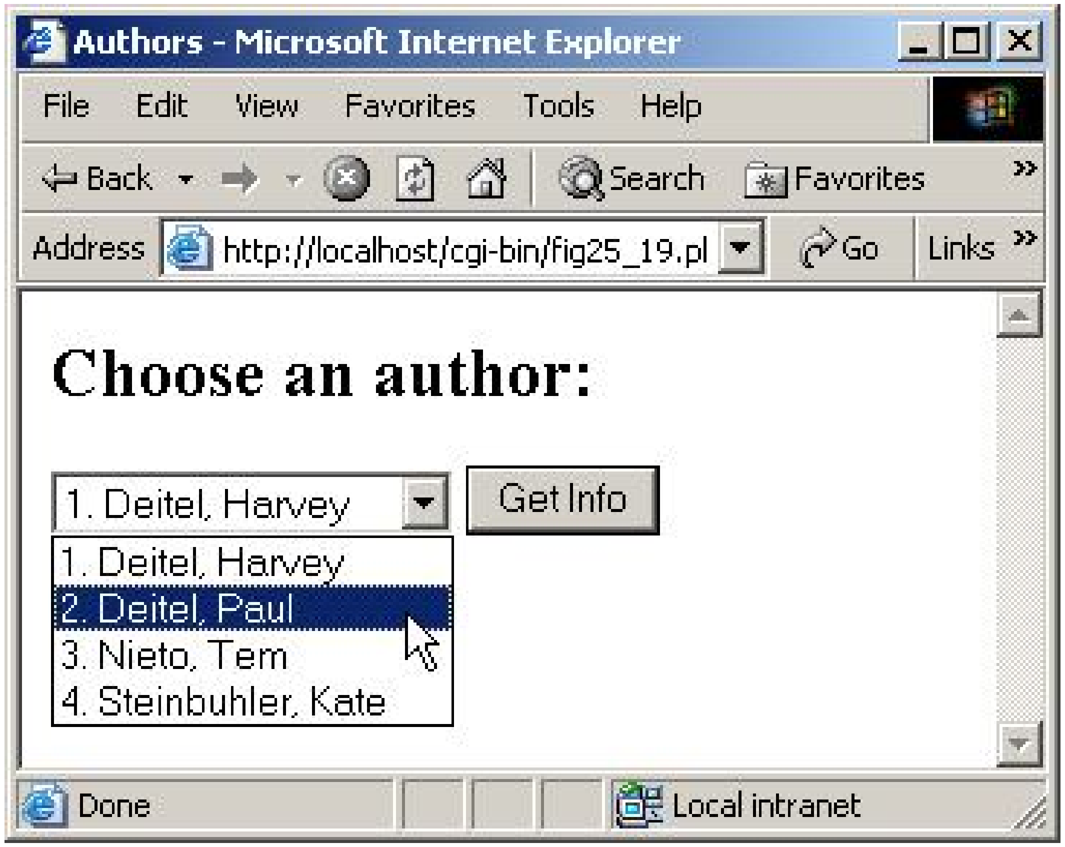
fig25_19.pl
(1 of 3)

```perl
# prepare the query for execution, then execute it
# a prepared query can be executed multiple times
$statementHandle = $databaseHandle->prepare( $query );
$statementHandle->execute();

print( h2( "Choose an author:" ) );

print( start_form( { action => 'fig25_20.pl' } ) );

print( "<select name = \"author\">\n" );

# drop-down list contains the author and ID number
# method fetchrow_array returns a single row from the result
while ( @row = $statementHandle->fetchrow_array() ) {
   print( "<option>" );
   print( "$row[ 2 ]. $row[ 1 ], $row[ 0 ]" );
   print( "</option>" );
}

print( "</select>\n" );

print( submit( { value => 'Get Info' } ) );
print( end_form(), end_html() );

```

fig25_19.pl
(2 of 3)

```
# clean up -- close the statement and database handles
$databaseHandle->disconnect();
$statementHandle->finish();
```

fig25_19.pl
(3 of 3)


Authors - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cgi-bin/fig25_19.pl
Go Links
Done
Local intranet

Choose an author:

1. Deitel, Harvey
Get Info

1. Deitel, Harvey
2. Deitel, Paul
3. Nieto, Tem
4. Steinbuhler, Kate

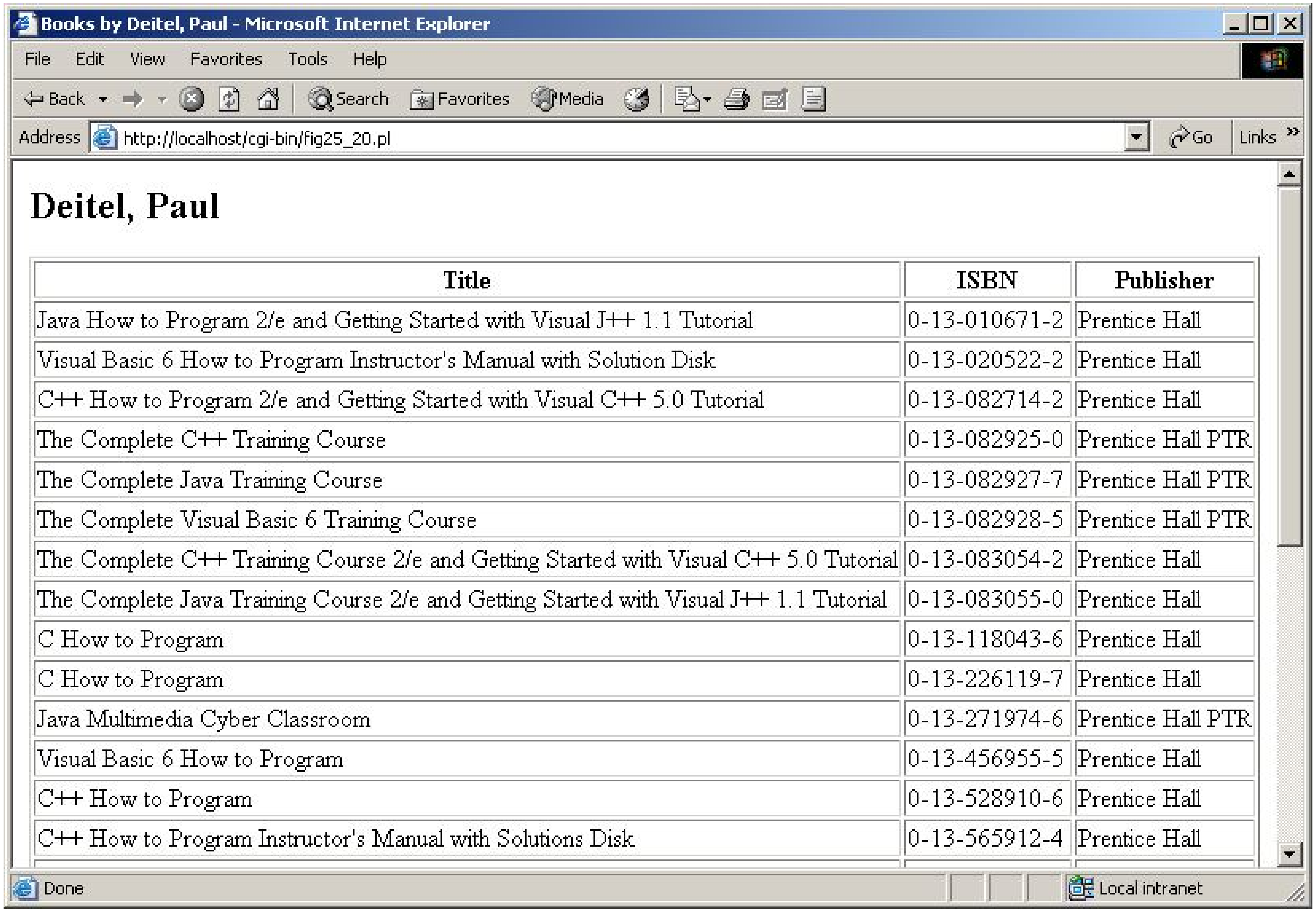
```perl
#!C:\Perl\bin\perl
# Fig. 25.20: fig25_20.pl
# CGI program to query a MySQL database.

use CGI qw( :standard );
use DBI;
use DBD::mysql;

$dtd =
"-//W3C//DTD XHTML 1.0 Transitional//EN\"
   \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

print( header() );

# retrieve author's ID and name from the posted form
$authorID = substr( param( "author" ), 0, 1 );
$authorName = substr( param( "author" ), 3 );

print( start_html( { dtd => $dtd,
                     title => "Books by $authorName" } ) );

$databaseHandle = DBI->connect( "DBI:mysql:books",
                     "root", "", { RaiseError => 1 } );

```

```perl
# use AuthorID to find all the ISBNs related to this author
$query1 = "SELECT ISBN FROM AuthorISBN
           WHERE AuthorID = $authorID";

$statementHandle1 = $databaseHandle->prepare( $query1 );
$statementHandle1->execute();

print( h2( "$authorName" ) );

print( "<table border = 1>" );
print( th( "Title" ), th( "ISBN" ), th( "Publisher" ) );

while ( @isbn = $statementHandle1->fetchrow_array() ) {
   print( "<tr>\n" );

   # use ISBN to find the corresponding title
   $query2 = "SELECT Title, PublisherID FROM titles
              WHERE ISBN = \'$isbn[ 0 ]\'";
   $statementHandle2 = $databaseHandle->prepare( $query2 );
   $statementHandle2->execute();
   @title_publisherID = $statementHandle2->fetchrow_array();

   # use PublisherID to find the corresponding PublisherName
   $query3 = "SELECT PublisherName FROM Publishers
              WHERE PublisherID = \'$title_publisherID[ 1 ]\'";
```

fig25_20.pl
(2 of 3)

```perl

   $statementHandle3 = $databaseHandle->prepare( $query3 );
   $statementHandle3->execute();
   @publisher = $statementHandle3->fetchrow_array();


   # print resulting values
   print( td( $title_publisherID[ 0 ] ), "\n" );
   print( td( $isbn[ 0 ] ), "\n" );
   print( td( $publisher[ 0 ] ), "\n" );

   print( "</tr>" );

   $statementHandle2->finish();
   $statementHandle3->finish();
}

print( "</table>" );

print( end_html() );

$databaseHandle->disconnect();
$statementHandle1->finish();
```

fig25_20.pl
(3 of 3)


Books by Deitel, Paul - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address http://localhost/cgi-bin/fig25_20.pl
Go Links
Done
Local intranet

Deitel, Paul

Title	ISBN	Publisher
Java How to Program 2/e and Getting Started with Visual J++ 1.1 Tutorial	0-13-010671-2	Prentice Hall
Visual Basic 6 How to Program Instructor's Manual with Solution Disk	0-13-020522-2	Prentice Hall
C++ How to Program 2/e and Getting Started with Visual C++ 5.0 Tutorial	0-13-082714-2	Prentice Hall
The Complete C++ Training Course	0-13-082925-0	Prentice Hall PTR
The Complete Java Training Course	0-13-082927-7	Prentice Hall PTR
The Complete Visual Basic 6 Training Course	0-13-082928-5	Prentice Hall PTR
The Complete C++ Training Course 2/e and Getting Started with Visual C++ 5.0 Tutorial	0-13-083054-2	Prentice Hall
The Complete Java Training Course 2/e and Getting Started with Visual J++ 1.1 Tutorial	0-13-083055-0	Prentice Hall
C How to Program	0-13-118043-6	Prentice Hall
C How to Program	0-13-226119-7	Prentice Hall
Java Multimedia Cyber Classroom	0-13-271974-6	Prentice Hall PTR
Visual Basic 6 How to Program	0-13-456955-5	Prentice Hall
C++ How to Program	0-13-528910-6	Prentice Hall
C++ How to Program Instructor's Manual with Solutions Disk	0-13-565912-4	Prentice Hall

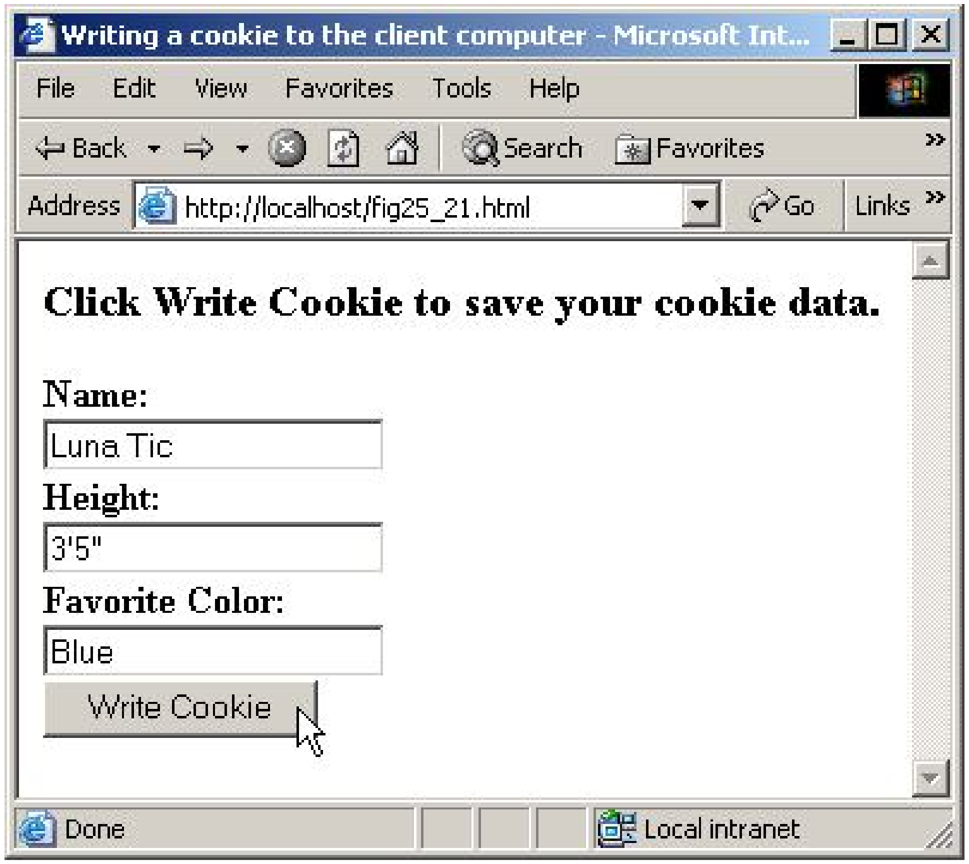
Cookies and Perl

- Cookies
  - Maintain state information
  - Function `time`
    - Returns current date and time
  - `SetCookie: header`
    - Indicate browser should store incoming data in cookie
  - `HTTP_COOKIE`
    - Contains client’s cookies

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- Fig. 25.21: fig25_21.html -->

<html>
   <head>
      <title>Writing a cookie to the client computer</title>
   </head>

    <body style = "font-face: arial">
         <div style = "font-size: large;
                       font-weight: bold">
            Click Write Cookie to save your cookie data.
         </div><br />

         <form method = "post" action = "cgi-bin/fig25_22.pl"
               style = "font-weight: bold">
            Name:<br />
            <input type = "text" name = "name" /><br />
            Height:<br />
            <input type = "text" name = "height" /><br />
            Favorite Color:<br />
            <input type = "text" name = "color" /><br />
            <input type = "submit" value = "Write Cookie" />
```

fig25_21.html
(1 of 2)

```
         </form>
      </font>
   </body>
</html>
```


Writing a cookie to the client computer - Microsoft Int...
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/fig25_21.html
Go Links
Done
Local intranet

Click Write Cookie to save your cookie data.

Name:

Luna Tic

Height:

3'5"

Favorite Color:

Blue

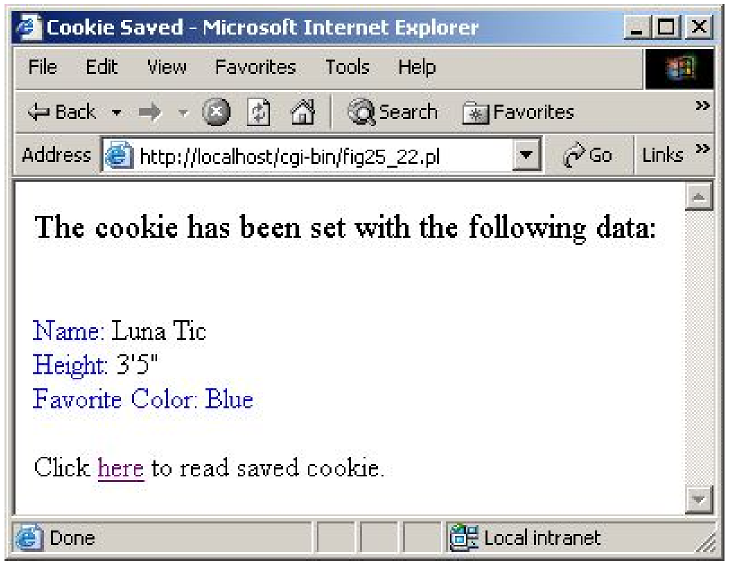
Write Cookie

```
#!C:\Perl\bin\perl
# Fig. 25.22: fig25_22.pl
# Program to write a cookie to a client’s machine.

use CGI qw( :standard );

$name = param( "name" );
$height = param( "height" );
$color = param( "color" );

 $expires = gmtime( time() + 86400 );

 print( "Set-Cookie: Name=$name; expires=$expires; path=\n" );
 print( "Set-Cookie: Height=$height; expires=$expires; path=\n" );
 print( "Set-Cookie: Color=$color; expires=$expires; path=\n" );

 $dtd =
 "-//W3C//DTD XHTML 1.0 Transitional//EN\"
    \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

 print( header() );
 print( start_html( { dtd => $dtd,
                      title => "Cookie Saved" } ) );

```

fig25_22.pl
(1 of 2)

```
print <<End_Data;
<div style = "font-face: arial; font-size: larger">
   The cookie has been set with the following data:
</div><br /><br />

<span style = "color: blue">
Name: <span style = "color: black">$name</span><br />
Height: <span style = "color: black">$height</span><br />
Favorite Color:</span>

<span style = "color: $color"> $color</span><br />
<br />Click <a href = "fig25_25.pl">here</a>
to read saved cookie.
End_Data

print( end_html() );
```

fig25_22.pl
(2 of 2)

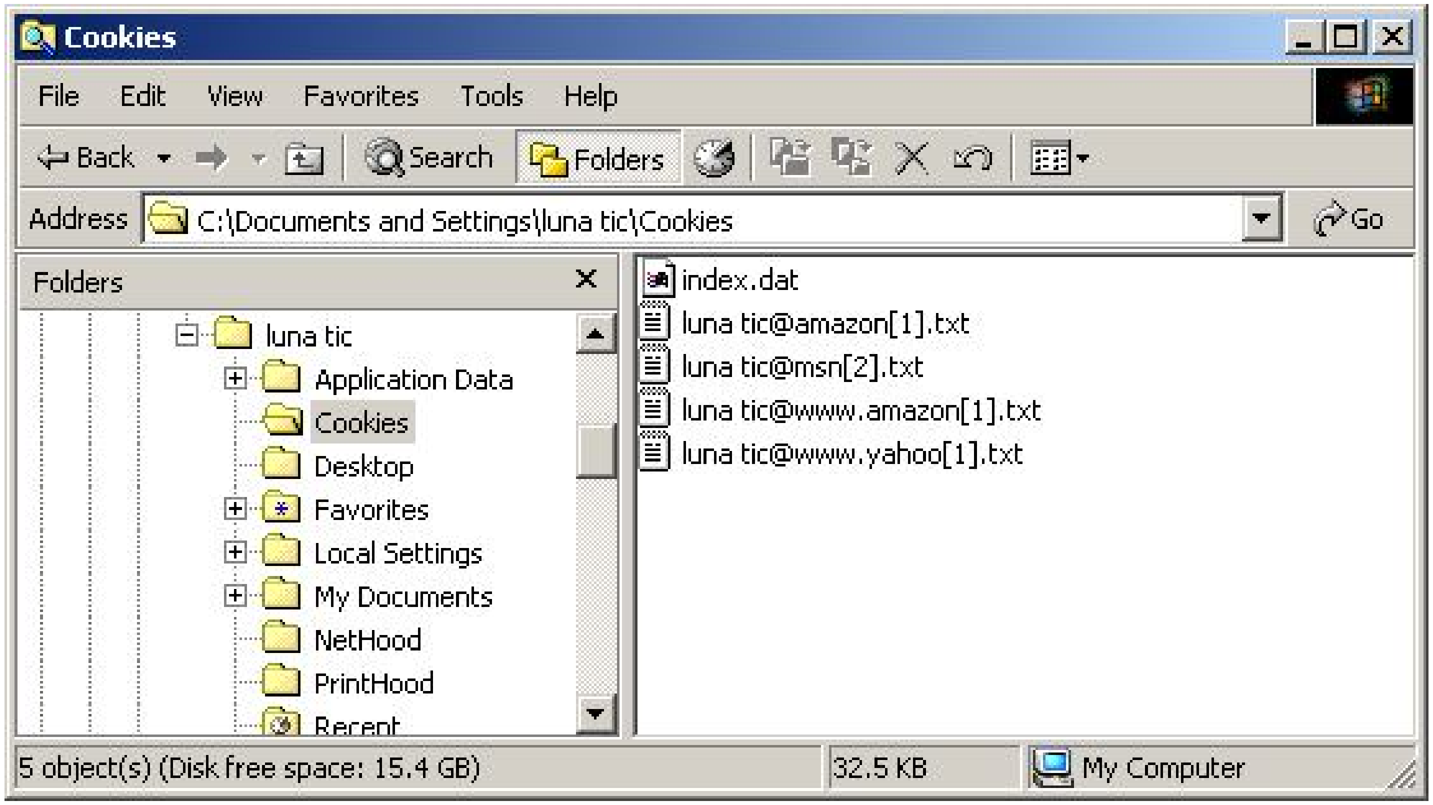

Cookie Saved - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address http://localhost/cgi-bin/fig25_22.pl Go Links
Done Local intranet

The cookie has been set with the following data:

Name: Luna Tic
Height: 3'5"
Favorite Color: Blue

Click here to read saved cookie.

Cookies and Perl



C:\Documents and Settings\luna tic\Cookies

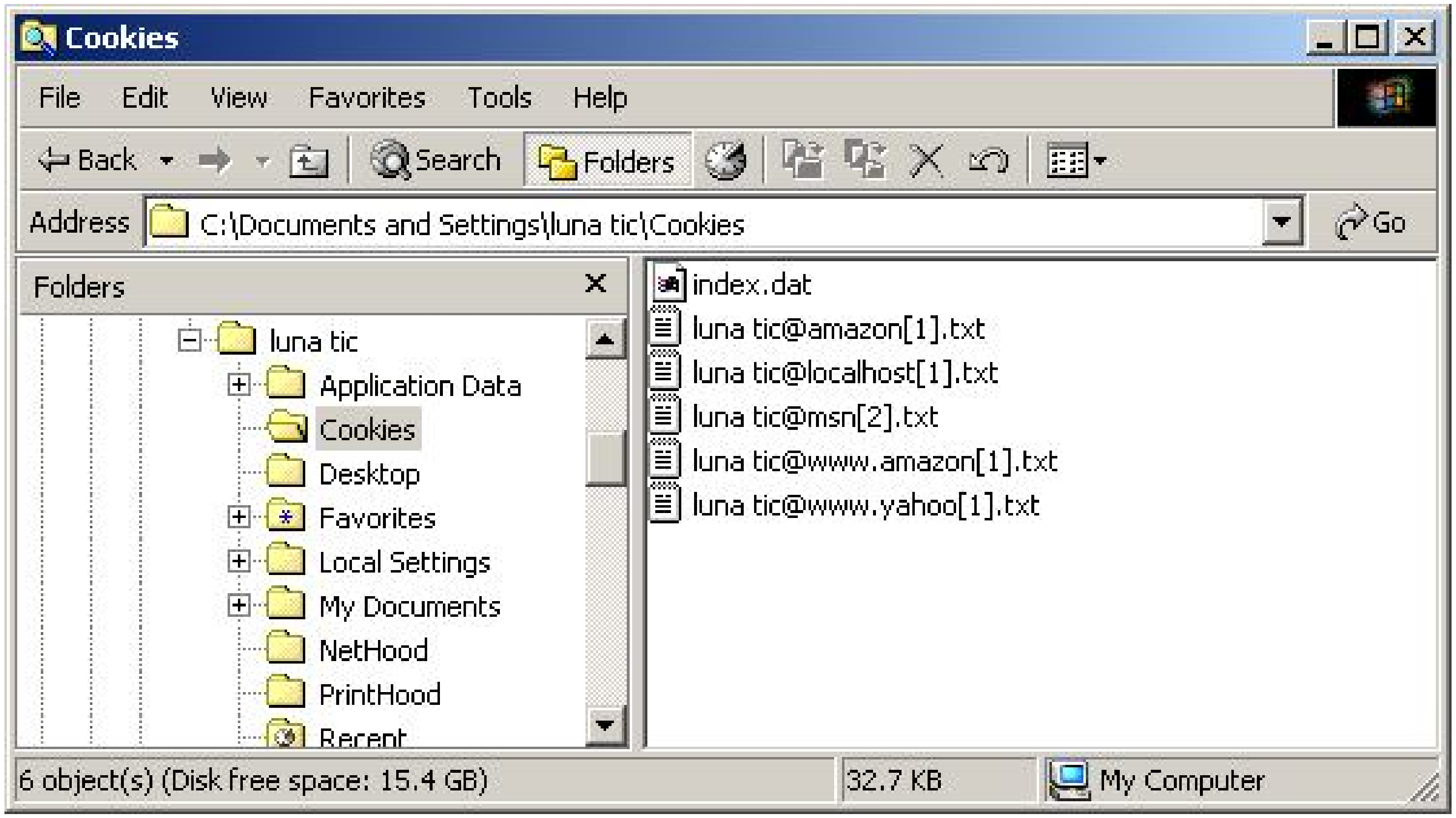
luna tic
Application Data
Cookies
Desktop
Favorites
Local Settings
My Documents
NetHood
PrintHood
Recent

index.dat
luna tic@amazon[1].txt
luna tic@msn[2].txt
luna tic@www.amazon[1].txt
luna tic@www.yahoo[1].txt

5 object(s) (Disk free space: 15.4 GB) 32.5 KB My Computer

Fig. 25.23 Cookies directory before a cookie is written.

Cookies and Perl



C:\Documents and Settings\luna tic\Cookies

luna tic
Application Data
Cookies
Desktop
Favorites
Local Settings
My Documents
NetHood
PrintHood
Recent

index.dat
luna tic@amazon[1].txt
luna tic@localhost[1].txt
luna tic@msn[2].txt
luna tic@www.amazon[1].txt
luna tic@www.yahoo[1].txt

6 object(s) (Disk free space: 15.4 GB) 32.7 KB My Computer

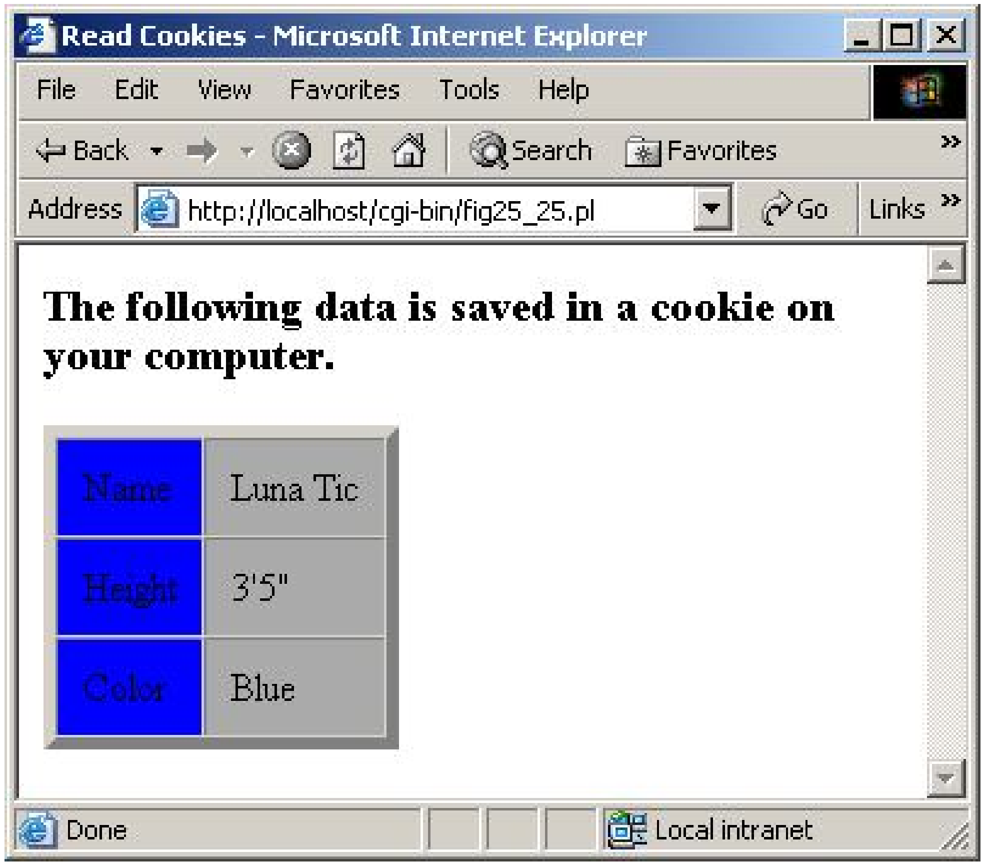
Fig. 25.24 Cookies directory after a cookie is written.

```perl
#!C:\Perl\bin\perl
# Fig. 25.25: fig25_25.pl
# program to read cookies from the client's computer.

use CGI qw( :standard );

$dtd =
"-//W3C//DTD XHTML 1.0 Transitional//EN\"
   \"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd";

print( header() );
print( start_html( { dtd => $dtd,
                     title => "Read Cookies" } ) );

print( div( { style => "font-face: arial;
                        font-size: larger;
                        font-weight: bold" },
          "The following data is saved in a
           cookie on your computer." ), br() );

print( "<table style = \"background-color: #aaaaaa\"
              border = 5 cellpadding = 10
              cellspacing = 0>" );

%cookies = readCookies();
```

fig25_25.pl
(1 of 2)

```perl
$color = $cookies{ Color };

foreach $cookieName ( "Name", "Height", "Color" ) {
   print( Tr( td( { style => "background-color: $color" },
                  $cookieName ),
              td( $cookies{ $cookieName } ) ) );
}

print( "<table>" );
print( end_html() );

sub readCookies
{
   @cookieArray = split( "; ", $ENV{ 'HTTP_COOKIE' } );

   foreach ( @cookieArray ) {
      ( $cookieName, $cookieValue ) = split( "=", $_ );
      $cookieHash{ $cookieName } = $cookieValue;
   }

   return %cookieHash;
}
```

fig25_25.pl
(2 of 2)


Read Cookies - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back
Search
Favorites
Address
Go
Links
Done
Local intranet

http://localhost/cgi-bin/fig25_25.pl

The following data is saved in a cookie on your computer.

Name	Luna Tic
Height	3'5"
Color	Blue

Operator Precedence Chart

Operator	Type	Associativity
terms and list operators	`print @array` or `sort (4, 2, 7)`	left to right
`->`	member access	left to right
`++` `--`	increment decrement	none
`**`	exponentiation	right to left

`!`
`~`
`\`
`+`
`-`
	logical NOT
	bitwise one’s complement
	reference
	unary plus
	unary minus
		right to left

`=~`
`!~`
	matching
	negated match
		left to right

`*`
`/`
`%`
`x`
	multiplication
	division
	modulus
	repetition
		left to right

`+`
`-`
`.`
	addition
	subtraction
	string concatenation
		left to right

Operator Precedence Chart

`<<`
`>>`
	left shift
	right shift
		left to right

`named unary operators`	unary operators—e.g., `-e` (filetest)	none

`<`
`>`
`<=`
`>=`
`lt`
`gt`
`le`
`ge`
	numerical less than
	numerical greater than
	numerical less than or equal to
	numerical greater than or equal to
	string less than
	string greater than
	string less than or equal to
	string greater than or equal to
		none

`==`
`!=`
`<=>`
`eq`
`ne`
`cmp`
	numerical equality
	numerical inequality
	numerical comparison (returns -1, 0 or 1)
	string equality
	string inequality
	string comparison (returns -1, 0 or 1)
		none

`&`	bitwise AND	left to right
`|` `^`	bitwise inclusive OR bitwise exclusive OR	left to right
`&&`	logical AND	left to right

Operator Precedence Chart

||	logical OR	left to right
..	range operator	none
?:	conditional operator	right to left
=	assignment	right to left
+=	addition assignment	
-=	subtraction assignment	
*=	multiplication assignment	
/=	division assignment	
%=	modulus assignment	
**=	exponentiation assignment	
.=	string concatenation assignment	
x=	repetition assignment	
&=	bitwise AND assignment	
|=	bitwise inclusive OR assignment	
^=	bitwise exclusive OR assignment	
<<=	left shift assignment	
>>=	right shift assignment	
&&=	logical AND assignment	
||=	logical OR assignment	

Operator Precedence Chart

`,`
`=>`
	expression separator; returns value of last expression
	expression separator; groups two expressions
		left to right

`not`	logical NOT	right to left
`and`	logical AND	left to right
`or` `xor`	logical OR logical exclusive OR	left to right

Fig. 25.26 Perl operator precedence chart.