

300+

IMMEDIATE SOLUTIONS FROM THE PROS

FLASH
AND GIF
MOTION

SECURE
E-COMMERCE
TRANSACTIONS

OBJECT
PLACEMENT
WITH
CASCADING
STYLE
SHEETS

GRAPHIC
IMAGE
MANAGEMENT

PHP-
GENERATED
WEB
CONTENT

WEB PAGE
OPTIMIZATION

DATABASE
FUNCTIONALITY
AND
INFORMATION
MANAGEMENT

INTERACTIVITY
WITH
HTML FORMS
AND
TABLES

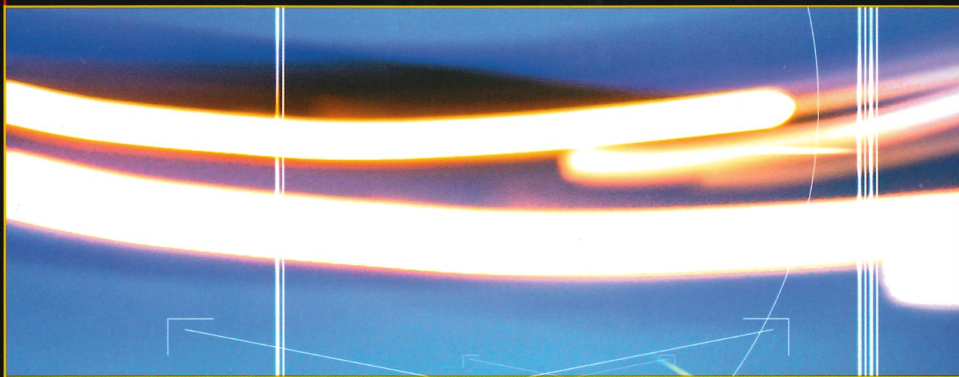
FAST
DOWNLOADS
WITH
FILE
COMPRESSION

SOUND,
VIDEO, AND
MULTIMEDIA

ACTIVEX
OBJECTS
AND
JAVASCRIPT

ONLINE
STORE
MANAGEMENT

HTML & Web Design



TIPS & TECHNIQUES

Kris Jamsa Ph.D., MBA

Best-selling author of more than 90 computer books

Konrad King

Enterprise database consultant and author

Andy Anderson

HTML and Web design curriculum developer and author

Mc
Graw
Hill

OSBORNE

3/07
FREE BOOKS
48 \$ 928

FREE
CODE
ONLINE

HTML & Web Design

Tips & Techniques

Kris Jamsa
Konrad King
Andy Anderson

McGraw-Hill/Osborne

New York Chicago San Francisco
Lisbon London Madrid Mexico City Milan
New Delhi San Juan Seoul Singapore Sydney Toronto

McGraw-Hill/Osborne
2600 Tenth Street
Berkeley, California 94710
U.S.A.

To arrange bulk purchase discounts for sales promotions, premiums, or fund-raisers, please contact **McGraw-Hill/Osborne** at the above address. For information on translations or book distributors outside the U.S.A., please see the International Contact Information page immediately following the index of this book.

HTML & Web Design Tips & Techniques

Copyright © 2002 by The McGraw-Hill Companies. All rights reserved. Printed in the United States of America. Except as permitted under the Copyright Act of 1976, no part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of publisher, with the exception that the program listings may be entered, stored, and executed in a computer system, but they may not be reproduced for publication.

1234567890 DOC DOC 0198765432

ISBN 0-07-219394-8

Publisher	Brandon A. Nordin
Vice President & Associate Publisher	Scott Rogers
Acquisitions Editor	Jim Schachterle
Project Editor	Janet Walden
Acquisitions Coordinator	Tim Madrid
Technical Editors	Charles Hornberger, Wendy Willard
Copy Editor	Darren Meiss
Proofreader	Pat Mannion
Indexer	David Heiret
Computer Designers	Carie Abrew, George Toma Charbak, Tabitha M. Cagan
Illustrators	Michael Mueller, Lyssa Wald
Series Designer	Roberta Steele
Cover Series Design	Greg Scott

This book was composed with Corel VENTURA™ Publisher.

Information has been obtained by **McGraw-Hill/Osborne** from sources believed to be reliable. However, because of the possibility of human or mechanical error by our sources, **McGraw-Hill/Osborne**, or others, **McGraw-Hill/Osborne** does not guarantee the accuracy, adequacy, or completeness of any information and is not responsible for any errors or omissions or the results obtained from the use of such information.

564 HTML & Web Design Tips & Techniques

```
<% Set objMySQLCon = Server.CreateObject("ADODB.Connection")
objMySQLCon.ConnectionString = _
"DRIIVER={MySQL}; SERVER=NVBizNet2;UID=root;PWD=;DATABASE=hwtdtt"
objMySQLCon.Open
%>
```

The connection object's *ConnectionString* property contains the values necessary for the script to log in to the DBMS. In this example, the *objMySQLCon.Open* method uses the MySQL ODBC driver to connect with the MySQL DBMS on the NT server named NVBizNet2. The connection logs in under the username (UID) root, which has no password, and elects to use the hwtdtt database.

After establishing a connection with the DBMS, use the connection object's *Execute* method to submit a query as shown here:

```
<%
querystring = _
"SELECT COUNT(*) Count FROM siteAccessList" & _
" WHERE username = '" & Request.Form("username") & "' & _
" AND password = '" & Request.Form("password") & "'"
Set objResultsSet = objMySQLCon.Execute(querystring)
If objResultsSet.Fields("Count") = 1 Then
Response.Redirect "/hwtdtt/MemberWelcome.asp"
End If
%>
```

In this example, the script uses the connection object (*objMySQLCon*) *Execute* method (*objMySQLCon.Execute*) to submit a query string (*querystring*) to the MySQL DBMS. The DBMS, in turn, executes the query and returns the rows of columns that match the search criteria (that is, the query's results set) within the *objResultsSet* object.

The query in this example will always return a single row of results in a column named *Count*. [*Count(*)* in the query submitted is an SQL aggregate function that returns the count of table rows that satisfy the search condition(s) in the query's WHERE clause.] If the DBMS finds a matching username/password pair in the *siteAccessList* table, the DBMS sets the *Count* field to "1", and the script redirects the Web browser to the Welcome page within the site's members-only area. Conversely, if either the username or the password typed into the login form is invalid, the DBMS returns zero (0) in the *Count* field. The script then moves on to HTML tags and text (not shown here), which prompts the visitor to log in again.

You can retrieve the Web page with the login form and the Active Server Page with the login authentication script described in this Tip by downloading the file Chpt11Tip09.zip from the Osborne Web site at <http://www.osborne.com>.

Note that setting a form element's type attribute to "password" obscures text the visitor enters into the element from view. (The Web browser displays an asterisk [*] in place of each character the visitor types into the element.) However, after the visitor clicks the Submit button, the Web browser sends all form data (including text typed into a password element) as unencrypted ASCII text. As such, a hacker could tap into the Internet and read (as plain text) the body of the HTTP message in which the Web browser placed the username and password it sent to the Web server.

To safeguard sensitive information (such as usernames and passwords), ask the ISP hosting your site to set up the Secure Socket Layer (SSL) protocol. The ISP must first obtain a *security certificate* for your Web site from a certificate authority such as VeriSign Corporation (<http://www.verisign.com>), Thawte Consulting (<http://www.thawte.com>), or Entrust (<http://www.entrust.com>). After the ISP installs the security certificate for your site on the Web server, the Web server and Web browser will encrypt their HTTP messages before sending them across the Internet. Having the Web browser and server encrypt their transmissions protects your form results (and other message content) from unauthorized access, because a hacker viewing an encrypted HTTP message body sees only an unintelligible jumble of characters, numbers, and special symbols.

SSL encryption and decryption occur outside the scope of your Active Server Page scripts—meaning you can use the same login (or other) script whether communicating across a secure or nonsecure connection. The only difference you will notice is that your secure URLs will start with <https://> instead of the standard (nonsecure) <http://>. Thus, to use a secure connection when sending form results to an Active Server Page script, you would use a URL like <https://www.NVBizNet2.com/hwtdtt/authenticate.asp>.

After receiving an encrypted HTTP request for an Active Server Page, the Web server decrypts the message and passes encrypted data (such as form results) along with the Web page requested to the ASP script host. The script host, in turn, calls on a script engine to execute the Active Server Page script(s) embedded within the Web page and sends the script output back to the Web server. The Web server, in turn, places the script-generated page into the body of an HTTP response, which the server then encrypts and sends to the Web browser. Upon receipt of the HTTP response (that is, the Web page requested from the Web server) the Web browser decrypts the message and renders the page.

Starting a Session and Working with Session Variables

An Active Server Page session is a continuous period during which a visitor interacts with a Web site. A session might consist of a single or a series of transactions and the visitor might retrieve one or several Web pages during the course of a single session. For example, during a session, a visitor might check the weather on a local news site. Or, the visitor might log in to a bank's Web site to check account balances, pay bills online, and transfer money between accounts. In short, an Active Server Page session begins when a visitor requests the first Active Server Page from a site and ends when the visitor leaves the site, when a script abandons the session, at the expiration of a timeout interval, or (immediately) if the visitor's browser refuses the session cookie.

Sessions are an important Web site design tool because they let Active Server Pages scripts use variables that maintain their state (that is, variables that keep their values) throughout the visitor's stay on the Web site. As a visitor moves from page to page on a site, you can use session variables to keep track of the visitor's personal data (such as name, address, and phone number), items the visitor puts into an online shopping cart, selections he or she makes on multipage forms, and so on. Although you could create persistent variables with cookies and hidden form elements, by using session variables, you reduce the amount of data that must pass between the Web server and browser with each HTTP request.

The Web server sends only one "key" value, the session ID, to the Web browser at the start of the session. The session ID is a cookie named ASPSESSIONID (or ASPSESSIONIDXXXXXXXX on