An introduction to ASP

Aslan Neishaboori

The history of ASP

- ASP -> Active Server Pages
- Microsoft’s server-side technology for dynamically-generated web pages that is marketed as a companion to Internet Information Server (IIS).

ASP from beginning until now

- ASP has gone through four major iterations
  - ASP 1.0 (distributed with IIS 3.0)
  - ASP 2.0 (distributed with IIS 4.0)
  - ASP 3.0 (distributed with IIS 5.0)
  - ASP.NET (part of the Microsoft .NET platform).
  - ASP.NET (1.0, 1.1, 2.0...)

Web pages

- Static web pages
- Client Side scripting
- Server side scripting / programming.

Static Web Pages

Client-side scripting, processing is done on client

Client requests web page: [http://www.calstatela.edu/faculty/csui6.html](http://www.calstatela.edu/faculty/csui6.html)

Server finds file and sends it to client (file1.html)

Client’s browser displays the HTML document on the screen

Client executes the script embedded in the file and displays result
Server-side scripting

ASP is an example of server-side scripting
In server side-scripting, processing is done at server rather than at client

- Server locates file on the server.
- Server executes the script in file.
- The output of the script is packaged in HTML format and sent back to client.

Client's browser displays the resulting HTML on the screen.

Advantages of Server-Side Scripting

- Control over server side components are available (access, SQL 2000, etc.); so, no client-side plug-ins required.
- Compatibility issues minimized (browser does not have to support client-side scripting).
- Bandwidth enhanced (send only the data the client needs).

One important Advantage of ASP

- Security enhanced
  - Users never see code
  - server is intermediary between client and data source

Security implemented

Some general points about ASP

- ASP technology is built directly into Microsoft Web servers, and therefore is supported on all Microsoft Web servers.
- ASP runs as a service of the Web server and is optimized for multiple threads and multiple users.

Some general points about ASP

- Web Pages can be generated by
  - mixing server-side scripting code (including database access)
  - with HTML
  - and client-side script.
- ASP is not a language it is a "server technology".

Many scripting languages can be used to write ASP

- VBScript (a subset of Microsoft Visual Basic programming language)
- Jscript (Microsoft's implementation of JavaScript)
- PerlScript—similar to Perl
- Python another scripting language used in web development.
What you need to run ASP

- You must be running a web server
  - Free web server products are available from Microsoft
  - Internet Information Server (IIS) requires NT Server or Win2k, Windows XP professional and above...
  - Personal Web Server (PWS) (stripped-down version of IIS) runs on Windows 95 or better

ASP Without IIS or PWS

- Several 3rd-party companies have created software to allow ASP to run on various non-Microsoft servers and platforms
  - Examples:
    - Haley Software’s Instant ASP
    - Childsafe’s CNVASP
  - These products extend ASP functionality to Non-MS Web Servers such as:
    - Apache, Sun Web Server, Netscape Enterprise Server
  - These products run on Non-MS platforms such as:
    - Linux, Sun Solaris, Apple Mac OS, IBM AIX

Creating ASP Pages

- File name must have .ASP extension
- Scripts are embedded within the HTML (but executed at Server)
- Start page off with @language directive:
  `<%@ Language=VBScript %>`
- Enclose script commands in "<%" and ">%"

Example:
```
<%Response.Cookies("MyFavMovie") = "Godfather"%>
```

An ASP example

```
<% If Hour(Now) < 9 Then %>
  Do you know what time it is? I was still in bed!
<% Else %>
  Randomize Int(Rnd*4)
  Select Case IntChoice
  Case 0
    So, where do you want me to go today?
  Case 1
    I came to help, look who’s back visiting us again!
  Case 2
    Hi there, and welcome to our site.
  Case 3
    It’s raining here – would you like to play virtual checker?
  End Select
<% End If %>
```

ASP Objects

- ASP includes five standard objects for global use:
  - Response: send info to user
  - Request: get info from user
  - Server: Controls the IIS server (i.e. creates objects and supplies access to methods and properties on the web server)
  - Session: stores session information for individual users as they navigate a web site
  - Application: stores and shares information for use during an active application

The Request object

- The Request object is used to get information from the user that is passed along in an HTTP request:
  - Form—to get data from an HTML form
  - Cookies—to get the value of application-defined cookie
  - ServerVariables—to get HTTP information such as the server name
Example

◆ Request.ServerVariables(Remote_Host)

◆ This example returns the Internet name or IP address of the visiting computer

Example

◆ Assume user has downloaded a form-based page from a site. The user selects an item in an Option drop-down list and hits the submit button.

  <FORM ACTION=
  "http://www.calstatela.edu/faculty/csun/grades.asp"
  METHOD=POST>

Example (Continued)

◆ The action calls the ASP page containing the following code:
   IF Request.Form("Answer") = "I agree"
      Response.Write "You may proceed"
   ELSE
      Response.Write "You cannot proceed"
   EndIf

The Response Object

◆ The Response object is used to send information to the user. The Response object supports a number of properties and methods.

Some Properties supported by Response Object

◆ ContentType—to set the type of content (i.e. text/HTML, Excel, etc.)
◆ Expires—sets the expiration (when the data in the user's cache for this Web page is considered invalid) based on minutes (i.e.: expires in 10 minutes).
◆ ExpiresAbsolute—allows you to set the expiration date to an absolute date and time.

Example

.....

  <H1>Welcome to the New Products Seminar</H1>
  <% Dim strCity, strDate
      strDate = Request.Form("Date")
      strCity = Request.Form("City")
      Response.Write "Held in ", strCity
      Response.Write " on ", strDate
      Response.Write %>

  ..
The following methods are supported by the Response object:

- **AddHeader**—Adds an HTML header with a specified value
- **AppendToFile**—Appends a string to the end of the Web server log file
- **BinaryWrite**—writes binary data (i.e., Excel spreadsheet data)
- **Clear**—clears any buffered HTML output.
- **End**—stops processing of the script.
- **Flush**—sends all of the information in the buffer.
- **Redirect**—to redirect the user to a different URL
- **Write**—to write into the HTML stream. This can be done by using the construct

  ```
  Response.Write("hello")
  ```

  or the shortcut command

  

---

### Session Object

- **Session object uses cookies to maintain state information**
- **To set session variables**

  ```
  Session("chrCity") = Request("chrCity")
  ```

---

### Working with Database Objects

- **Makes use of the ActiveX Data Objects Component (ADO)**
- **Two important ADO objects:** Connection and Recordset
  - Connection object is used to make the connection to the database
  - Recordset performs two tasks:
    - Instructs database as to what information you’re interested in
    - Stores the requested information returned by the database

---

### Database Objects in Subscription

- **Create an ADO database connection**
  ```
  set dbSubs = server.createobject("adodb.connection")
  ```

- **Open the connection using our ODBC file DSN**
  ```
  dbSubs.open("filedsn=SubForm")
  ```

- **Execute the SQL statement**
  ```
  dbSubs.execute(sql)
  ```

---

### Resources:

- **Example codes are from Beginning ASP 3.0 provided at www.wrox.com**