



Introduction to Active Server Pages

What is ASP?

ASP is not a language. It is not really an application, like FrontPage 98 or Word 97,

ASP is a technology for building dynamic and interactive web pages.

Active Server Pages is a programming environment that provides the ability to combine HTML, scripting, and components to create powerful Internet applications that run on your server.

Shortcomings of HTML

While browsing the web, you have probably noticed that the pages on many sites are composed simply of text and static images. These pages are usually composed almost entirely of HTML (with perhaps a little JavaScript code in some cases). These pages allow you to click on links or images to get to other pages, but these display much the same sort of content. Innovations, such as frames and tables, have helped to improve the presentation and usability of these sites, but they haven't made the crucial step towards making these pages truly **dynamic**.

You will find that sites that contain ASP code are more dynamic. They are quite often tailored to the individual user, they can reflect the fact that a user has visited the site before, they can be customized easily to view preferred topics, and in general they offer the user an all round richer experience.

The importance of Web Server

So what is Web server? It's simply a computer that provides web services on the Internet, or on a local intranet. A basic web designed to locate, address, and send out simple HTML pages. The web server makes web pages available to all other users, who can access these pages via their local network or via the Internet.

Enter CGI

As a simple layout language, HTML is a without peer, but as an application engine, it leaves a great deal to be desired. Help quickly appeared in the form of Common Gateway Interface (CGI) applications. A CGI application runs entirely on the server. When a browser contacts the server and makes a file request, the CGI application returns HTML just as if it were an HTML file. The difference is that the CGI program can process information sent by the browser and return HTML that differs in response to varying conditions. CGI programs can send one type of information to one user, and another type of information to another user, or send different files based on the type and version of browser the user is running.

Although you can write CGI programs in almost any language, most early CGI programmers used PERL, a language with powerful text-processing capabilities. A scripting language is a relatively small, usually interpreted (not compiled) language. These CGI programs used HTML forms to accept search requests from users. Unfortunately, CGI programs had several shortcomings. The Web is a world of small transactions. Each transaction consists of a single request or a series of short requests by the browser. When you request a page, the Web server supplies the page, then forgets about you. For busy sites such requests may happen at the rate of 100 requests per second or more. The first generation of CGI programs consisted of executables that had to be loaded into memory for each request. The CGI program would then process the request and terminate. All of this loading and unloading used up time and resources on the server. It obviously takes considerably more memory and CPU time to load a program and execute it then it does to simply return the contents of an HTML file. Web server designers began to look for a more efficient method to deliver dynamic content.

ISAPI

The concept they standardized on is called Internet Server Application Programming Interface (ISAPI). Unlike CGI programs, these new ISAPI programs remained loaded in memory in the server's address space. Thus ISAPI programs could perform the same tasks as CGI programs, but without the overhead of loading and terminating the program for each request. Officially, ASP is an ISAPI application.

What happens to HTML Pages?

When a request for a page from the browser, the web server, and the web server performs three steps;

- Reads the request from the browser
- Finds the page in the server
- Sends the page back across the internet to the browser

What happens to ASP Pages?

Instead of throwing a static HTML page out to the user, we want the server to take some actions according to our ASP code: the ASP will make some decisions and create a page that is tailored for that particular user. Thus, when using ASP, the server actions are as follows:

- Reads the request from the browser
- Find the page in the server
- Perform any instructions provided in ASP to modify the page
- Send the page back across the internet to the browser

Whenever a user enters a URL into a Browser's address field, clicks a link, or submits a form, the browser packages up information about itself, the URL, and (in some cases) the user, and sends that information to the sever as a Web request. The ASP engine consists of a collection of objects that contain information about the request, the scripting technology used to make decisions about how to handle the request (the code), and the Web server itself. Before you can understand much about what the ASP engine can do, you need to know what happens before the user's request reaches the code in your ASP page.

A Web request requires two components: a Web server and a client. The client is usually a browser, but could be another type of program.

Both the server and the client must use a defined protocol to communicate with each other. A protocol is simply an agreed-upon method for initiating a communications session, passing information back and forth, and terminating the session. There are several protocols used for Web communications; the most common are **Hyper Text Transfer Protocol (HTTP)** and **File Transfer Protocol (FTP)**. Regardless of the protocol used, Web requests are carried over an underlying network protocol called **Transmission Control Protocol / Internet Protocol (TCP / IP)**, which is a global communications standard that determines the rules two computers follow when they exchange information.

When you use a browser to retrieve an HTML page from a Web site, you are using the Hypertext Transfer Protocol (HTTP). The HTTP specifies how messages can be transported over the Internet. In particular, the protocol specifies the ways in which a browser and a Web server can interact. When a page is retrieved from a Web site, the browser opens a connection to a Web server at the Web site and issues a request. The Web server receives the request and issues a response. For this reason, the HTTP is called a request and response protocol.

The server computer runs an endless loop to check for communication initialization. The client sends an initialization to begin a session. The initialization is a defined series of bytes. When the server receives the initialization request, it acknowledges the transmission by returning another series of bytes to the client.

What can ASP do that HTML can't

The crucial difference is that pure HTML is interpreted by the browser, not executed on the server. By writing code that is to be executed on web server, you can achieve many more things than would otherwise be impossible.

- o Make it easier to edit contents of a web page, by updating a text file or the contents of a database rather than the HTML code itself
- o Create pages that will be customized to display only things that will be of interest to a particular user
- o Display and update databases contained in the web page, and manipulate the data therein, by being able to sort the entries into any order or view a subset of them
- o Create pages that rotate through a series of different graphics
- o Get feedback from a user, and return information to the user based upon that feedback.

ASP lets you use the power of a Web server to process user requests and provide dynamic, individualized, content based on logic file, and database data, and also process the user's individualized data. In other words, ASP lets multiple users simultaneously run a program on your Web server.

ASP provides:

- A way to save individualized data for each user
- Access to the file system
- Access to databases
- A means to launch and control any Component Object Model (COM) component

ASP and HTML

ASP is designed to be used together with HTML to create dynamic pages. In fact, ASP actually creates HTML code. Having said that, we really need to slow this discussion right up because the thought of text, HTML tags and ASP code, all intermingled and intermarried in a single piece of code, gives rise to one of the main problems for beginners.

A web page that uses ASP is likely to consist of a mixture of three types of syntax. Some of the page will be constructed from simple text, part will be the HTML, and part will be ASP code. The following table summarizes each of these ingredients.

Type	Purpose	Interpreter	Hallmarks
Text	Information to be shown on the page	Viewer's browser on their PC shows the text	Simple ASCII text
HTML tags	Instructions to the browser about how to format text and display images	Viewer's browser on their PC interprets the tags to format the text	Each tag within < > delimiters Usually has open and close tags, such as <TABLE>, </TABLE>
ASP statement	Instructions to the web server running ASP about how to create portions of the page to be sent out	Web site host's web server software with ASP extensions performs the instructions of the ASP code	Each ASP section contained within <% %> delimiters. ASP statements have a flavor of Visual Basic, with the appearance of programming code with variables, decision trees, etc

The primary difference between ASP and the other new generation technologies mentioned is that ASP must be executed on the web server, while the pages generated by other technologies are interpreted by the browser (or client). And the advantages that ASP enjoys over CGI and PERL are those of simplicity and speed.

Scripting Languages

In order to add depth to the capability provided by the HTML language, we can sprinkle our HTML code with commands that doesn't strictly belongs to HTML at all. Instead, these commands are written in one of a number of scripting languages that are available. We distinguish these "foreign" commands, embedded within the HTML code, by referring to them as scripts. Subsequently, when the page is viewed by the browser, each script is sent to a script host (an application that can run a program in another language), where it is interpreted by a script engine. Each scripting language needs its own interpreter to interpret it, so a VBScript program must be sent to a VBScript interpreter, and a JavaScript program to a JavaScript interpreter. Internet Explorer 4 contains both interpreters for VBScript and for Jscript (Microsoft's version of JavaScript), while Netscape Navigator 4 only contains a JavaScript Interpreter. ASP supplies interpreters (script engines) for both VBScript and Jscript

VBScript: Visual Basic Scripting Edition (VBScript) is the default scripting language of Active Server Pages. VBScript extends HTML with variables, operators, loops, conditionals, functions and subroutines. VBScript is a scripting language. A scripting language is typically easier to use when creating simple programs. VBScripts can be directly embedded into the HTML files. This allows you to extend HTML into something more than a page-formatting language. Other scripting languages like JScript, Perl or REXX can also be used with ASP

Client-Side or Server-Side Scripts?

So for the most part, our model of the browser making a connection, sending a request, receiving a reply and then getting the browser to interpret the subsequent HTML to construct a web page still holds true. The only bit that differs is that when the browser comes across something within script tags, it is submitted to the appropriate script engine for interpretation and execution. However (and here's the most important bit). The script host containing the script engine doesn't have to resident on the browser: it can be resident on either client-side, or server-side. The essential difference is as follows:

- A script that is interpreted by the browser is called a client-side script. A client-side script is an instruction set that is processed by the client, without contacting a server.
- A script that is interpreted by the web server is called a server-side script. A server-side script is an instruction set that is processed by the server, and the resulting data is sent to a client.

Client-Side Scripts

Client-side scripting involves the execution of the scripting language by the browser that interprets the web page. The main disadvantage of client-side scripting is that exactly how the script works is dependent on the type of browser that execute the script. In other words, client-side scripting is Browser specific.

Client-Side Programming Language

A client-side programming language is a language that can be interpreted and executed by a browser. When a program written in any of these languages is loaded into a compatible browser, the browser will automatically execute the program. Java and JScript/JavaScript are additional examples of client-side programming language.

The advantage of client-side programming language is that browsers do all the work. This places less of a burden on the server. Client-side programs can also be much faster than server-side programs.

Server-Side Programming Language A server-side programming executes on the server that serves a Web site's file, rather than on the browsers. It performs all the work on the Web site's computer and more burden on the server. The advantage of using VBScript is that the scripts work regardless of the browser in use. The scripts are processed before the pages are sent out across the Internet to browsers. Web browsers receive nothing more than normal HTML files. The easiest way to add a script to an ASP is by using script delimiters <% and %>. Any text enclosed within these delimiters will be processed as a script.

Introduction to ASP Objects

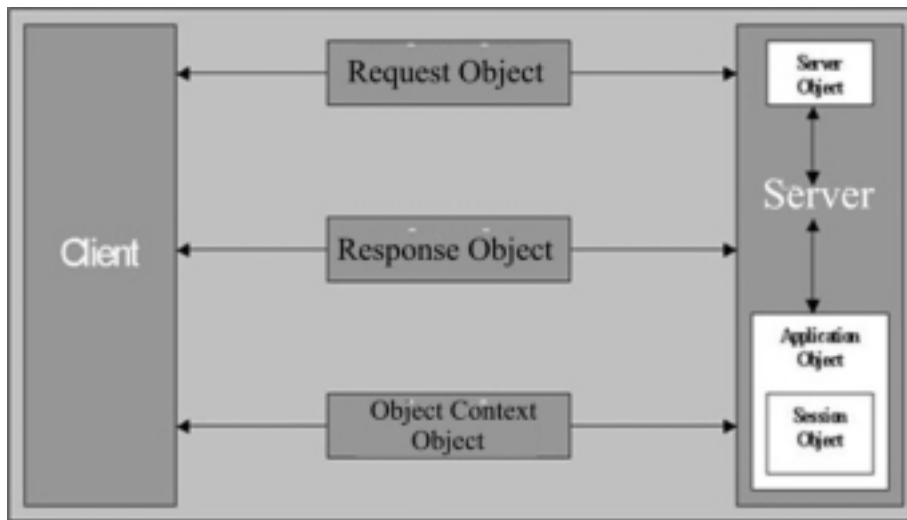
An object is nothing more than a set of properties and methods all wrapped up together with some property values. A property is a value that belongs to the object. Another way to think about properties is that they describe the state of an object. For example, a door object may have an Open property. The property may take only two values- True or False- either the door is open or it's closed. In Addition to properties, objects also have methods. A method differs from a property because a method does something whereas property is something. Typically, methods cause the object to do something- often based on its state. In other words, the methods of an object and the state of the object interact with each other and are often mutually dependant on each other.

To use most objects from a programming language, you must first create an instance of the object you want to use. An object instance is a single copy of an object. You create an instance of an object by creating an object variable, then assigning a new copy of an object to the variable. The process of creating an object instance is called instantiation. You instantiate an object when you create an instance of that object.

You don't have to instantiate ASP objects. The seven objects that ASP exposes are called intrinsic objects because you don't have to create them, they're always available for you to use on any ASP page. The objects are:

1. **Response Object:**
Used to send information to the client
2. **Request Object:**
Used to retrieve information included with the request from the client
3. **Server Object**
Used to communicate with the server
4. **Application Object:**
Used to store (cache) information about your application
5. **Session Object:**
Used to cache information about a specific browser instance(which usually, but not always, corresponds to a single user)
6. **ObjectContext Object:**
Used to initiate and control transactions and create new objects through Microsoft Transaction Server (MTS)

ASP- Object Model



Collection : A collection object contains a list of items. The items may or may not be in order, and may or may not be related to one another. ASP collections are all essentially identical in that they consist of a list of names, called keys, associated with values. Each key is associated with a single variant value (note that the value may be another list). For example, an ASP collection might look like this:

```
LastName="Doe"
FirstName="John"
Age="33"
```

This collection has three keys (Lastname, Firstname and age) and three values ("Doe", "John", and 33)

ASP collections all share three properties and methods:

- **Count :** The number of items in the collection. The first item (unlike an array) has an index of 1, not 0. To retrieve the count, use the syntax `Variable= CollectionName.Count`
- **Item:** This method lets you retrieve an item by name or index number. To retrieve an item, use the syntax `Variable = CollectionName.Item("Key-Name")` or `Variable=CollectionName.Item(Index)`.
- **Key:** This method lets you retrieve a key value by index. In practice, you will rarely use this method. Use the syntax `variable=CollectionName.Key`.

All ASP collection (and most other VB and VBScript collection objects) have a default property, the item property. In contrast to the Count or Key properties, you don't have to supply the item keyword to access the property.

The Request object contains five different collections of information.

Cookies : contains cookie values sent by the browser

Form : contains information the user enters into input controls and information your application has stored in form variables.

QueryString : contains information sent with the URL

ServerVariables : contains information that the server automatically parses for every request.

ClientCertificates : contains security information