ASP

تکنولوژی ASP متعلق به مایکروسافت چارچوبی برای ایجاد صفحات وب پویا فراهم می‌کند. با ASP می توانید اطلاعات داده شده در فرم‌ها را بازیابی کنید یا صفحات وب را برای یک کاربر خاص به صورت سفارشی ایجاد کنید.

ASP چیست؟

تکنولوژی ASP (Active Server Pages) متعلق به میکروسافت چارچوبی برای ایجاد صفحات وب پویا فراهم می‌کند که اینترنت و برنامه‌های کاربردی اینترنت را قادر می‌سازد باهم تعامل کنند.

با ASP می توانید محتوای یک صفحه وب را به طور پویا تغییر دهید، به پرس و جوهای کاربر یا داده های ارسال شده توسط فرم ها پاسخ بدهید، به داده های درون یک پایگاه داده دسترسی پیدا کنید و آنها را برای مرورگر ارسال کنید، صفحات وب را برای یک کاربر خاص به صورت سفارشی ایجاد کنید. بعلاوه چون کدهای ASP توسط مرورگرها قابل رویت نیستند امنیت بیشتر می شود. درضمن با طراحی هوشمندانه برنامه های ASP از ترافیک شبکه هم کاسته خواهد شد.

ASP برنامه‌ای است که درون IIS اجرا می‌شود. برای اجرای IIS کامپیوتر شما باید سیستم عامل Windows NT 4.0 یا نسخه‌های بعد از آن را دارا باشد. در این حالت کامپیوتر شما بعنوان وب سرور عمل می‌کند.

قبل از فراگیری ASP باید مفاهیم پایه‌ای HTML/XHTML و یک زبان اسکریپتی مانند JavaScript یا VBScript را بلد باشید.

فایل ASP

یک فایل ASP می تواند حاوی متن، تگ‌های HTML و اسکریپت باشد. اسکریپت‌های درون فایل ASP بر روی سرور اجرا می‌شوند.

کلیه صفحات ASP باید با پسوند ".asp" ذخیره شوند.

وقتی کاربر یک فایل HTML را از سرور درخواست می کند فایل موردنظر برای مرورگر ارسال می‌شود. اما وقتی یک فایل ASP را درخواست می کند، IIS درخواست را به موتور ASP می‌دهد تا فایل را خط به خط بخواند و اسکریپت‌های درون آنرا اجرا کند. در نهایت فایل به صورت HTML محض برای مرورگر ارسال می‌شود.

چون اسکریپت‌های ASP روی سرور اجرا می‌شوند مرورگری که آنها را نمایش می‌دهد نیازی به پشتیبانی از آنها ندارد.

چگونه ASP به IIS متصل می شود

ASP در حقیقت یک DLL به نام asp.dll است که به طور پیش فرض در فهرست Winnt\System32\intsrv نصب می‌شود. این DLL مسئول گرفتن صفحات ASP (که با انشعاب asp. مشخص شده‌اند) و بررسی آنها برای اسکریپت‌های سمت سرور است. این اسکریپت‌ها سپس به موتور اسکریپتی مناسب داده می‌شوند. نتایج اجرای اسکریپت با متن و HTML موجود در صفحه ترکیب می‌شود. صفحه کامل به وب سرور داده می شود که به نوبه خود آنرا به کاربری که صفحه را درخواست کرده می‌دهد.

بنابراین یک صفحه وب قبل از ارسال به سمت کاربر تغییر یا به طور پویا ایجاد می‌شود.

اسکریپت‌های سمت سرور می‌توانند عبارت، دستورات، زیربرنامه یا عملگرهای مجاز زبان اسکریپتی که انتخاب کردید باشند.

اسکریپت‌های سمت سرور توسط علائم <% و %> محصور می‌شوند. موتور ASP هر متنی که داخل آن باشد را تحلیل می‌کند.

ASP.NET چیست؟

تکنولوژی قدیمی صفحات سمت سرور ASP کلاسیک نامیده می‌شود. که بالاترین نسخه آن ASP 3.0 است. باوجود قدرت و انعطاف پذیری کدنویسی زیادی لازم بود درضمن یک قالب یا محیط برای پیاده سازی را در اختیار قرار نمی داد. ASP.NET راه حلی برای حل این مشکلات بود.

ASP.NET تکنولوژی کاملا جدیدی از شرکت میکروسافت برای صفحات سمت سرور است. در حقیقت نسل بعدی ASP است. ASP.NET بخش عمده چارچوب دات نت (.NET Framework) است.

چارچوب دات نت، زیربنای دانت نت و محیطی برای ایجاد، توسعه و اجرای برنامه‌ها و سرویس‌های وب است.

ASP+ نام اولیه ASP.NET بود که توسط میکروسافت استفاده می‌شد. ASP.NET 2.0، ASP.NET 3.0 و ASP.NET 4.0 نسخه های بعدی دات نت هستند که ابزارها و کتابخانه های جدیدی را به دات نت اضافه کردند.

دات نت نیز بر روی IIS اجرا می‌شود.

اشیای Response و Request

در ASP درخواست کاربر و پاسخ سرور از طریق دو شی Request و Response ساخته می‌شود.

تقریبا تمام کارهائی که در صفحه ASP انجام می‌دهیم بستگی به این دو شی دارد. نحوه استفاده از آنها می‌تواند روی کارائی و قدرت صفحات تاثیر بگذارد. البته کار اصلی آنها دسترسی به مقادیری است که کاربر از طریق قسمت <form> صفحه HTML یا اضافه کردن یک رشته به URL به سرور ارسال می‌کند و ایجاد خروجی مناسب برای برگرداندن به کاربراست.

شی Request شامل کلیه اطلاعات یا فرمی است که کاربر هنگام درخواست صفحه آماده کرده است. که شامل متغیرهای HTTP، cookie ها، کنترل‌های HTML در بخش فرم و هر مقدار اضافه شده به URL مثل QueryString است. همچنین هر گواهی که توسط SSL یا پروتکل‌های دیگر ارتباطی برای رمزگذاری استفاده می‌شود و خواصی که به ما در مدیریت این اتصال کمک می‌کند.

شی Response برای تولید پاسخی است که سرور برای ارسال به کاربر آماده می‌کند. که شامل متغیرهای HTTP، cookie ها، اطلاعاتی درباره محتوائی که به مرورگر ارسال می‌کنیم، و مجموعه‌ای از متدهای برای ایجاد خروجی مثل Response.Write است.

ورودی کاربر می‌تواند توسط دستور Request.Form یا Request.QueryString بازیابی شود.

Response.Write

دستور response.write برای نوشتن خروجی روی برازر است.

مثال. متن "Hello World" برای برازر ارسال می‌شود و در بدنه مستند نوشته می‌شود.

```
<html>
<body>
<%
  response.write("Hello World!")
%>
</body>
</html>
```

به جای دستور response.write یک راه کوتاهتر هم وجود دارد.

مثال. متن "Hello World" به برازر ارسال می‌شود.

```
<html>
<body>
<%
  ="Hello World!"
%>
</body>
</html>
```

نکته. زبان اسکریپتی پیش فرض ASP زبان VBScript است. ASP به شما امکان انتخاب زبان را می‌دهد. می‌توانید زبان را به صورت زیر تعیین کنید. اگر می خواهید زبان دیگری به عنوان زبان پیش‌فرض در یک صفحه خلص باشد در بالای صفحه مشخص کنید.
نکته. دقت داشته باشید JavaScript حساس به متن است. کدهای شما هرجا که لازم است باید با حروف بزرگ یا کوچک نوشته شوند.

مثال. زبان JavaScript به عنوان زبان پیش‌فرض تعریف شده است.

```
<%@ language="javascript"%>
<html>
<body>
<%
  Response.Write("Hello World!")
%>
</body>
</html>
```

مثال. از تگ‌های HTML برای فرمت خروجی می‌توان استفاده کرد.

```
<html>
<body>
<%
  response.write("<h2>You can use HTML tags to format the text!</h2>")
  response.write("<p style='color:#0000ff'>This text is styled with the style
attribute!</p>")
%>
</body>
</html>
```

مثال. در VBScript علامت تک کوتیشن برای شروع خط توضیحات استفاده می‌شود.

```
<%@ language="vbscript"%>
<%
  'This is a comment.
  'Nothing after a tick will be seen by the user
%>
```

Response.Redirect

متد redirect کاربر را به صفحه دیگری هدایت می کند.

مثال. هدایت مرورگر کاربر به آدرس http://www.hpkclasses.ir.

```
<%
  Response.Redirect "http://www.hpkclasses.ir"
%>
```

متغیرهای ASP

کلیه متغیرها در ASP به صورت Variant تعریف می‌شوند. نیاز به تعریف آنها به صورت عدد یا رشته نیست.

مثال. اسکریپت زیر محتوای دو متغیر را به همراه پیغام در برازر نشان می دهد.

```
<%
  Dim SomeText
  Dim SomeNum

  SomeText = "ASP is great!"
  SomeNum = 1

  Response.Write(SomeText)
  Response.Write(SomeNum)

  Response.Write ("Right now you are thinking that " & SomeText)
%>
```

مثال. اعلان یک آرایه برای ذخیره اسامی.

```
<%
  Dim famname(5),i
  famname(0) = "Jan Egil"
  famname(1) = "Tove"
  famname(2) = "Hege"
  famname(3) = "Stale"
  famname(4) = "Kai Jim"
  famname(5) = "Borge"

  For i = 0 to 5
      response.write(famname(i) & "<br>")
  Next
%>
```

نکته. متغیری که خارج از زیربرنامه تعریف شود توسط هر اسکریپتی درون فایل ASP قابل دسترسی و تغییر است.
نکته. متغیرهایی که درون زیربرنامه تعریف می شوند هنگام خروج از زیربرنامه از بین می روند. اسکریپتی خارج از زیربرنامه به آنها دسترسی ندارد.
نکته. برای تعریف متغیرهایی که در بیشتر از یک فایل قابل دسترسی باشند از متغیرهای session و application استفاده می‌شود.

شیء Session

از شی Session برای ذخیره و بازیابی اطلاعات و مقادیر هر کاربر در صفحه استفاده می‌شود.

پروتکل HTTP یک پروتکل ناپایدار است یعنی سرور هر درخواست وب را به صورت یک در خواست مستقل پردازش می‌کند و اطلاعی درباره متغیرها و اطلاعاتی که در درخواست‌های قبلی استفاده شده‌اند ندارد. شی Session ذخیره و بازیابی اطلاعات مورد نظر به ازای هر کاربر را هنگام حرکت بین صفحات برنامه وب میسر می‌کند.

متغیرهایی که در شیء Session ذخیره می‌شوند اطلاعاتی درباره یک کاربر می‌دهند و در کلیه صفحات یک برنامه وب قابل دسترس هستند.

سرور برای هر کاربر جدید یک شیء Session ایجاد می‌کند . درخواست‌های ارسالی از سوی یک کاربر تا زمانی که پنجره مرورگرش باز و فعال است شناسایی و اطلاعات لازم و مرتبط در Session قرار می‌گیرد.

مثال. در تکه کد زیر نحوه ایجاد دو متغیر با نام‌های username و age است که به ترتیب با "ali" و "۵۰" مقداردهی شده‌اند.

```
<%
  Session("username")="ali"
  Session("age")=50
%>
```

مثال. روش دوم مقداردهی به متغیرهای Session.

```
<% Session.Add("name", "Ali");%>
```

مثال. مقدار متغیر name از Session خوانده و در صفحه وب به نمایش در می آید.

```
Welcome <% Response.Write(Session("username"))%>
```

مثال. حذف متغیر sale در صورتی که متغیر age زیر 18 باشد.

```
<%
  If Session.Contents("age")<18 then
    Session.Contents.Remove("sale")
  End If
%>
```

مثال. اگر بخواهیم تمای اطلاعات موجود در Session شامل تمامی متغیرها آن را پاک کنیم از تابع RemoveAll به شکل زیر استفاده می‌کنیم.

```
<%
  Session.Contents.RemoveAll()
%>
```

مثال. برای این که بخواهیم تمامی مقادیر موجود در متغیرهای Session را بخوانیم مانند تکه کد زیر عمل می‌کنیم.

```
<%
  dim i
  dim j
  j=Session.Contents.Count
  Response.Write("Session variables: " & j)
  For i=1 to j
    Response.Write(Session.Contents(i) & "<br>")
  Next
%>
```

تا زمانی که Session منقضی نشده باشند می توانیم متغیرهای آنرا بازیابی و استفاده کنیم. زمانی که کاربر هیچ درخواست صفحه‌ای از برنامه ارسال نکند، Sessionها خاتمه یا به اصطلاح منقضی (expired) می‌شود. این زمان به شکل پیش‌فرض ۲۰ دقیقه می‌باشد. البته این زمان قابل تغییر است و بسته به حساسیت و میزان امنیت برنامه کاربردی توسط برنامه نویس می‌تواند مقداردهی شود.

خاصیت Timeout در شیء Session برای تنظیم زمانی غیر از پیش‌فرض است.

مثال. فاصله زمانی ۵ دقیقه برقرار می‌شود.

```
<%
  Session.Timeout=5
%>
```

مثال. متد Abandon برای خاتمه فوری یک session است.

```
<%
  Session.Abandon
%>
```

نکته. پیدا کردن این فاصله زمان انقضا ساده نیست. تشخیص اینکه آیا درخواست کاربر آخرین بوده یا خیر راحت نیست.
نکته. تنها مقدار کمی اطلاعات را در متغیرهای Session ذخیره کنید.

شیء Application

یک برنامه کاربردی روی وب معمولا از چندین صفحه ASP تشکیل شده که با هم کار می‌کنند. شیء Application برای متصل کردن این فایل‌ها به یکدیگر است. متغیرهای Application مشابه Session در کلیه صفحات یک برنامه کاربردی قابل دسترسی هستند با این تفاوت که کلیه کاربران از یک شیء مشترک بهره می‌برند. به همین دلیل برای نگهداری اطلاعاتی که در همه صفحات بکار می‌روند نظیر اتصال پایگاه‌داده استفاده می‌شوند.

اگر در جایی شیء Application تغییر کند روی کلیه صفحات تاثیر می‌گذارد.

مثال. ایجاد متغیر Application.

```
<script language="vbscript" runat="server">

Sub Application_OnStart
application("vartime")=""
application("users")=1
End Sub

</script>
```

مثال. نمایش متغیر مثال بالا.

```
There are
<%
  Response.Write(Application("users"))
%>
active connections.
```

مثال. مجموعه Contents کلیه متغیرهای برنامه را در خود دارد. کد زیر با یک حلقه محتوای آنرا نمایش می‌دهد.

```
<%
  dim i
  For Each i in Application.Contents
    Response.Write(i & "<br>")
  Next
%>
```

نکته. با متد Lock در شیء Application می‌توانید یک برنامه را قفل کنید. بقیه کاربران قادر به تغییر متغیرهای Application برنامه قفل شده نخواهند بود. با متد Unlock قفل باز می‌شود.

Request.QueryString

کلیه مقادیری که کاربر با پرکردن <form> درصفحه آماده می‌کند یا آنهائی که به انتهای URL اضافه می‌شوند توسط دو مجموعه Form و QueryString بازیابی می‌شوند.

دستور Request.QueryString برای جمع آوری مقادیر در فرمی است که با صفتخاصه "method="get ساخته شده است. اطلاعاتی که با متد get ارسال می‌شوند برای همه قابل‌رویت است زیرا در بخش آدرس مرورگر نشان داده می‌شود.

مثال. نمونه یک فرم ورودی که با متد get ارسال می‌شود.

```
<form action="simpleform.asp" method="get">
      FirstName: <input name="fname" type="text">
      LastName:  <input name="lname" type="text">
      <input type="submit" value="send">
</form>
```

اگر در فرم بالا کاربر مقادیر "Bill"و "Gates" را وارد کند آدرس URL که برای سرور ارسال می‌شود به صورت زیر خواهد بود.

```
http://www.MyWebSite.com/simpleform.asp?fname=Bill&lname=Gates
```

اگر فایل"simpleform.asp" شامل اسکریپت زیر باشد.

```
<body>
Welcome
<%
  response.write(request.querystring("fname"))
  response.write(" " & request.querystring("lname"))
%>
</body>
```

مرورگر متن زیر را نمایش خواهد داد:

Welcome Bill Gates

نکته. مجموعه مقادیری که می توان با متد get ارسال کرد محدود است.

Request.Form

برای گردآوری اطلاعات درون یک فرم که توسط متد "post" ارسال شده از دستور Request.Form استفاده می شود. اطلاعاتی که با متد post ارسال می‌شوند برای بقیه قابل رویت نیستند و برخلاف متد get محدودیتی در مقدار ندارند.

مثال. اگر در فرم بالا کاربر مقادیر "Bill"و "Gates" را وارد کند آدرس URL که برای سرور ارسال می‌شود به صورت زیر است.

```
http://www.MyWebSite.com/simpleform.asp
```

اگر فایل"simpleform.asp" شامل اسکریپت زیر باشد مرورگر خروجی مشابه مثال قبل را خواهد داشت.

```
<body>
Welcome
<%
  response.write(request.form("fname"))
  response.write(" " & request.form("lname"))
%>
</body>
```

مثال. می‌توانیم با دانستن ایندکس صحیح کنترل روی فرم (که از کنترل اول در HTML شروع می‌شود و به ترتیب تعریف کنترل‌ها ادامه پیدا می کند) به مجموعه فرم دسترسی پیدا کنیم. البته این حالت توصیه نمی شود چون ترتیب کنترل‌ها ممکن است تغییر کند.

```
<%
  FirstName=request.form(1))
  LastName=request.form(2))
%>
```

مثال. کل مجموعه مقادیر فرم بدون کلید یا ایندکس بازیابی می شود.

```
AllFormContent=Request.Form
```

مثال. هدایت کاربر به صفحه دیگر

```
<%
if Request.Form("select")<>"" then
      Response.Redirect(Request.Form("select"))
end if
%>
<html>
<body>

<form action="demo_redirect.asp" method="post">

<input type="radio" name="select"
value="demo_server.asp">
Server Example<br>

<input type="radio" name="select"
value="demo_text.asp">
Text Example<br><br>
<input type="submit" value="Go!">

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

مثال. ایجاد یک لینک تصادفی

```
<html>
<body>

<%
randomize()
r=rnd()
if r>0.5 then
  response.write("<a href='http://www.w3schools.com'>W3Schools.com!</a>")
else
  response.write("<a href='http://www.refsnesdata.no'>Refsnesdata.no!</a>")
end if
%>

<p>
This example demonstrates a link, each time you load the page, it will display
one of two links: W3Schools.com! OR Refsnesdata.no! There is a 50% chance for
each of them.
</p>

</body>
</html>
```

مثال. کنترل بافر

```
<%
Response.Buffer=true
%>
<html>
<body>
<p>This is some text I want to send to the user.</p>
<p>No, I changed my mind. I want to clear the text.</p>
<%
Response.Clear
%>
</body>
</html>
```

مثال. پاک کردن بافر

```
<%Response.Buffer=true%>
```

```
<html>
<body>
<p>This is some text I want to send to the user.</p>
<p>No, I changed my mind. I want to clear the text.</p>
<%
Response.Clear
%>
</body>
</html>
```

مثال. پایان دادن به اسکریپت در میان پردازش و برگرداندن نتایج

```
<html>
<body>
<p>I am writing some text. This text will never be<br>
<%
Response.End
%>
finished! It's too late to write more!</p>
</body>
</html>
```

مثال. بررسی اینکه کاربر هنوز متصل است یا خیر

```
<%
If Response.IsClientConnected=true then
Response.Write("The user is still connected!")
else
Response.Write("The user is not connected!")
end if
%>
```

مثال. تنظیم مجموعه کاراکتر

```
<%
Response.Charset="ISO8859-1"
%>
<html>
<body>

<p>This is some text</p>

</body>
</html>
```

مثال. نمایش پیغام وابسته به زمان روی سرور.

```
<%
dim h
h=hour(now())

response.write("<p>" & now())
response.write("</p>")
If h<12 then
   response.write("Good Morning!")
else
   response.write("Good day!")
end if
%>
```

مثال. مانند بالا

```
<%
var d=new Date()
var h=d.getHours()

Response.Write("<p>")
Response.Write(d)
```

```
Response.Write("</p>")
if (h<12)
   {
   Response.Write("Good Morning!")
   }
else
   {
   Response.Write("Good day!")
   }
%>
```

مثال. نمایش رشته Query string

```
<%
Response.Write(Request.QueryString)
%>
```

مثال. چگونه یک صفحه چند دقیقه قبل از انقضا در مرورگر ذخیره شود.

```
<%Response.Expires=-1%>
<html>
<body>
<p>This page will be refreshed with each access!</p>
</body>
</html>
```

مثال. تنظیم زمان/تاریخ وقتی یک صفحه در مرورگر ذخیره می شود.

```
<%
Response.ExpiresAbsolute=#May 05,2001 05:30:30#
%>
<!DOCTYPE html>
<html>
<body>
<p>This page will expire on May 05, 2001 05:30:30!</p>
</body>
</html>
```

مثال. استفاده از یک حلقه برای تیترهای مختلف در HTML.

```
<%
dim i
for i=1 to 6
   response.write("<h" & i & ">Heading " & i & "</h" & i & ">")
next
%>
```