

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

مقالات

ASP

ورود اطلاعات فارسی با استفاده از جاوااسکریپت

اگر قبلا اقدام به استفاده از زبان فارسی در کدهای خود و مخصوصا هنگام کار با Microsoft Access کرده باشید به احتمال زیاد با مشکلات موجود بر سر راه آشنا هستید. یکی از مشکلات اساسی بر سر راه استفاده از زبان فارسی در بانکهای اطلاعاتی Microsoft Access در نحوه ورود اطلاعات می باشد. برای درک بهتر این موضوع اینگونه عمل نمایید:

در ویندوز خود قابلیت فارسی نویسی را فعال نمایید و پس از آن در Access يك Table با نام test و با يك فیلد به نام test از نوع text ایجاد نمایید. سپس با استفاده از خود Access اطلاعاتی به زبان فارسی وارد نمایید. سپس از Access خارج شده و با استفاده از ASP به بانک اطلاعاتی Access متصل شده و اطلاعات وارد شده را خوانده و نمایش دهید. مشاهده خواهید نمود که اطلاعات نمایش داده شده داخل مرورگر با اطلاعات وارد شده توسط شما در داخل Access متفاوت می باشد.

برای حل این مشکل چه باید کرد؟

برای حل این مشکل باید اطلاعات از خارج از Access و با استفاده از ASP و يك تابع Javascript که صفحه کلید را در همه ویندوزها اعم از فارسی و انگلیسی، فارسی نموده و امکان تایپ فارسی را فراهم میکند، وارد شوند. روش کار به این صورت است که در يك فرم HTML و در داخل يك Input Box هنگام فشردن شدن هر کلیدی از صفحه کلید، تابع Javascript فرا خوانده شده و معادل فارسی حرف وارد شده را نمایش میدهد. برای نمایش حروف فارسی لازم است که از Times New Roman و یا Tahoma انتخاب نمایید. این دو فونت فونتهای استاندارد ویندوز بوده و برای دیده شدن متون فارسی نیازی به نصب فونت جدید روی سیستم کاربر نخواهد بود. Javascript استفاده شده در زیر آمده است:

```
گ،(و-%$#@<br>0123456789/.'ch<br>arCodeAt(key-32);<br>else if (e<br>e.which='<br>گ،(و-%$#@<br>0123456789/.'ch<br>arCodeAt(key-32);<br>}<br>return true;<br>}<br>//--><br></SCRIPT>
```

کتابخانه: حروف فارسی در اینجا پشتیبانی نشده است

پس از طراحی فایل HTML نوبت به ایجاد فایل ASP می رسد. در فایل ASP باید اطلاعات وارد شده در فرم HTML را دریافت نموده و در بانک اطلاعاتی Access توسط برنامه خود بنویسید. نکته جالب این است که اگر وارد Microsoft Access شوید و اطلاعات وارد شده توسط برنامه خود را مشاهده نمایید اثری از حروف فارسی مشاهده نخواهید کرد. حال اگر دوباره توسط برنامه ASP اولیه به بانک اطلاعاتی متصل شوید و اطلاعات دریافت شده را داخل مرورگر نمایش دهید اطلاعات به صورت فارسی نمایش داده میشوند. لازم به ذکر است که باید از charset=windows-1256 در فایل ASP که اطلاعات فارسی را نمایش میدهد استفاده نمایید.

یک شمارنده ساده یا کونتور

این برنامه یک مثال خوب از نحوه باز کردن ، خواندن و نوشتن داخل یک فایل است که نیاز به هیچ نوع بانک اطلاعاتی هم ندارد . این فایل را با اسم counter.asp ذخیره کنید و در کنارش یک فایل متنی ساده به کمک NotePad و به نام database.inc ایجاد کنید و در سطر اولش هم عدد صفر یا هر عدد دیگری را که مایلید پیش فرض شمارنده شما باشد بنویسید. فراموش نکنید که سطح دسترسی لازم برای نوشتن را نیز به این فایل بدهید . با کمی ابتکار می توان این شمارنده را به نوع گرافیکی تبدیل کرد، حتی می توانید به تعداد صفحات سایت ، فایل ایجاد کنید تا تعداد مشاهده هر صفحه را به تنهایی ردیابی کنید.

```
<%  
Dim objFSO , objFile  
set objFSO = createobject("scripting.filesystemobject")  
set objFile = objFSO.opentextfile(server.mappath("database.inc"))  
'get the current count and store in counter  
counter = cLng(objFile.readline)  
'add one to the current count  
counter = counter + 1  
'close file object  
objFile.close  
'write a new text object with the same name and new count  
Set objFile = objFSO.CreateTextFile(server.mappath("database.inc"), true)  
objFile.writeline(counter)  
'close file object  
objFile.close  
,  
  
counter_length = len(counter)  
'loop through the hit count and display the image  
for i = 1 to counter_length  
%>  
<%= mid(counter,i,1) %>  
<%  
next  
%>
```

ارسال ایمیل در ASP ساده و کار آمد

با این کد ساده می توان صفحه ای برای ارسال ایمیل ایجاد کرد . این کد را یک هندی به نام مستعار Prince Joseph نوشته است و آنطور که گفته ملزومات سیستمی آن برای اجرا عبارتند از : یک ویندوز نسخه سرویس دهنده و SMTP Service بر روی IIS . این دو صفحه با استفاده از کامپوننت یا ابزار CDO که توسط شرکت مایکروسافت ارائه شده کار می کنند .

: mailSend.htm

```
<html><body>
<form id=frmMail method=post action=mailSend.asp>
TO:<input type=text name=txtTo size=30><br>
FROM:<input type=text name=txtFrom size=30><br>
SUBJECT:<input type=text name=txtSubject size=30><br>
BODY:<textarea rows="5" name=txtBody cols="28">
</textarea><br>
<input type=submit value="Send the Mail">
</form>
</body></html>
```

: mailSend.asp

```
<%@ Language=VBScript %>
<%
Dim objMail
Set objMail = CreateObject("CDONTS.NewMail")
objMail.From = Request.Form("txtFrom")
objMail.To= Request.Form("txtTo")
objMail.Subject= Request.Form("txtSubject")
objMail.Body= Request.Form("txtBody")
objMail.Send
Set objMail = Nothing
Response.Write "Mail should have been sent"
%>
```

درج تاریخ شمسی

تا به حال الگوریتم‌های زیادی برای نمایش تاریخ جاری بر اساس تقویم جلالی ارائه شده‌اند . در زیر یک نمونه ساده ولی کارآمد آن ارائه می شود . این نمونه تاریخ Server را نشان می دهد . برای اینکه تاریخ از روی کامپیوتر بازدیدکننده محاسبه شود کافیست runat=server را از کد حذف کنید .

```
<script language="vbscript" runat="server">
Function IranDate()
Dim ifday,ifmonth,ifyear,ifdayofyear
Dim iyear,idayofyear
Dim inumdayofyear
Dim aifmonthdays
aifmonthdays = Array(31,31,31,31,31,31,30,30,30,30,30,29)
inumdayofyear = 365
iyear =year(date)
idayofyear = datepart("y", date)
If isleapyear(iyear-1) then
inumdayofyear = 366
aifmonthdays(11) = 30
End if
If (idayofyear > 79 ) then
ifyear = iyear -621
ifdayofyear = idayofyear - 79
Else
ifyear = iyear - 622
ifdayofyear = (inumdayofyear - 79 ) + idayofyear
End if
ifday = ifdayofyear
While ( ifday > aifmonthdays(ifmonth) )
ifday = ifday - aifmonthdays(ifmonth)
ifmonth = ifmonth + 1
Wend
ifmonth = ifmonth + 1
IranDate = ifyear & "/" & ifmonth & "/" & ifday
End Function
Function isleapyear(nyear)
isleapyear = ( ( nyear mod 4) = 0 And (nyear mod 100) <> 0 ) or (nyear mod 400) = 0 )
End Function
Sub Showfdate
End sub
</script>

<%= (IranDate) %>
```

فرستادن فایل یا Upload از صفحه وب

در اینجا به ذکر نمونه ای از يك کد آماده برای بارگزاری فایل از طریق صفحه وب (Upload) به يك سرویس دهنده می پردازیم . این کدها از مجموعه موجود در سایت گرفته شده اند . در ابتدا باید هر دو فایل زیر را بسازید :

: upload_page.asp

```
<!-- #include file="upload_class.asp" -->
<%
Server.ScriptTimeout = 300 'now i can upload and save files upto ~8Mb
Dim intLevel, intUpload, intSave, strError, strContentType, strFilename, lngFileSize
Dim objUpload
Dim lngTime, lngUploadTime, lngSaveTime
intLevel = Request.QueryString("level")
'-----
wrHead
If intLevel = 1 Then
    Set objUpload = New FileUpload
    With objUpload
        .Path = "D:\Inetpub\wwwroot\test"
        lngTime = Timer()
        intUpload = .Upload
        lngUploadTime = Round((Timer() - lngTime) * 1000,3)
        lngTime = Timer()
        intSave = .Save(true)
        lngSaveTime = Round((Timer() - lngTime) * 1000,3)
        strError = .Error
        strFilename = .Filename
        lngFilesize = .Size
        strContentType= .ContentType
    End With
    Set objUpload = Nothing
End If

wrForm
wr "<hr style='\"'height:1px;width:100%;\"' />"
wr "Upload = " & intUpload & "<br />"
wr "Save = " & intSave & "<br />"
wr "Error = " & strError & "<br />"
wr "Filename = " & strFilename & "<br />"
wr "Filesize = " & lngFilesize & "<br />"
wr "Content-Type = " & strContentType & "<br />"
wr "Upload time = " & lngUploadTime & " ms<br />"
wr "Save time = " & lngSaveTime & " ms <br />"
wr "<hr style='\"'height:1px;width:100%;\"' />"
wrFoot
'-----
```

```

Sub wrForm
    wr "<form method=""post"" enctype=""multipart/form-data""
action=""?level=1"">"
    wr "<input type=""file"" name=""file""></input>"
    wr "<input type=""submit""></input>"
    wr "</form>"
End Sub

```

```

Sub wrHead
    wr "<html>"
    wr "<head>"
    wr "<title>upload</title>"
    wr "</head>"
    wr "<body>"
End Sub

```

```

Sub wrFoot
    wr "</body>"
    wr "</html>"
    Response.End
End Sub

```

```

Sub wr(byval sText)
    If sText <> "" Then Response.Write sText & vbNewLine
End Sub
%>

```

: upload_class.asp

```

<%
'|-----+
'|Class:   FileUpload   |
'|Date:    11:01 PM 7/23/2002|
'|By:      M.Meijer     |
'|Version: 1.0           |
'|-----+
'|
'|To upload and save a file submitted within a html form
'|
'|**Remarks:
'|Uploading files with this class is not recommended for huge files,
'|it takes alot of time saving the file to a textstream (as it the function 'save' does).
'|It takes 7.812ms to upload a file from 'localhost', with a size of 40,000 bytes.
'|Saving this file however costs 1078.125ms, and it takes 145828.1ms to save a file of
'|5.5Mb.
'|Conclusion don't save big files, use the maxfilesize property to limit the filesize.

```

'The class can only handle one file on a submission.

'The file will be saved in the specified 'Path', if there is no 'path' set, it can't save the file. (doh!)

'Properties:

'-----

'	ContentType	string	read	Content-Type of the file
'	Filename	string	read/write	Name of the file
'	Path	string	read/write	A path to a directory with permissions to write the file
'	Size	long	read	The size of the file in bytes
'	AllowedFiles	string	read/write	Allowed file extension(s), multiple separated with a comma
'	Maxfilesize	long	read/write	Maximum allowed size of the file
'	Error	string	read	The explanation of an error if occurred

'Methods

'-----

'	Upload()	= Status		
'				Copies the result of Request.Binaryread to a file
'	Status	integer	0	Upload success
'			1	A file has not been posted
'			2	File exceeds the maximum allowed filesize
'			3	Type is not allowed
'	Save(Overwrite)	= Status		
'				Slaat de bytearray op in een bestand met de in Filename gedefinieerde bestandsnaam,
'				in de in Path gedefinieerde directory.
'	Overwrite	boolean	true	If the file exists it will be overwritten
'			false	If the file exists it will not be overwritten
'	Status	integer	0	The file has been saved
'			1	The binary value could not be written to a file
'			2	There is no binary value
'			3	The filename is empty
'			4	An error already occurred, can't continue

'Code:

'-----

Class FileUpload

```
Private strContentType  
Private bytData  
Private strFilename  
Private strPath  
Private lngTotalbytes  
Private strAllowedFiles  
Private lngMaxFileSize  
Private strError
```

```
Private Sub Class_initialize()  
    strContentType = ""  
    bytData = ChrB(10)  
    strFilename = ""  
    strPath = ""  
    lngTotalbytes = 0  
    strAllowedFiles = ""  
    lngMaxFileSize = 0  
    strError = ""  
End Sub
```

```
Private Sub Class_Terminate()  
    bytData = Null  
End Sub
```

```
Public Property Get Size  
    Size = lngTotalbytes  
End Property
```

```
Public Property Let MaxFileSize(byval vData)  
    If IsNumeric(vData) > 0 Then  
        lngMaxFileSize = vData  
    End If  
End Property
```

```
Public Property Get MaxFilesize  
    MaxFilesize = lngMaxFileSize  
End Property
```

```
Public Property Let AllowedFiles(byval vData)  
    If Len(vData) > 0 Then  
        strAllowedFiles = vData  
    End If  
End Property
```

```
Public Property Get AllowedFiles  
    AllowedFiles = strAllowedFiles  
End Property
```

```
Public Property Get Error
```

```
    Error = strError
```

```
End Property
```

```
Public Property Get ContentType
```

```
    ContentType = strContentType
```

```
End Property
```

```
Public Property Let Path(byval vData)
```

```
    If Len(vData) > 0 Then
```

```
        strPath = vData
```

```
    End If
```

```
End Property
```

```
Public Property Get Path
```

```
    Path = strPath
```

```
End Property
```

```
Public Property Let Filename(byval vData)
```

```
    If Len(vData) > 0 Then
```

```
        strFilename = vData
```

```
    End If
```

```
End Property
```

```
Public Property Get Filename
```

```
    Filename = strFilename
```

```
End Property
```

```
Public Function Upload()' as integer
```

```
    Dim bytAllData
```

```
    lngTotalbytes = Request.Totalbytes
```

```
    If lngTotalbytes > 0 Then
```

```
        If lngMaxFileSize <> 0 Then
```

```
            If lngTotalBytes > lngMaxFileSize Then
```

```
                strError = "The file exceeds the allowed capacity."
```

```
                Upload = 2
```

```
                Exit Function
```

```
            End If
```

```
        End If
```

```
        bytAllData = Request.BinaryRead(lngTotalbytes)
```

```
        strContentType = GetContentType(bytAllData)
```

```
        strFilename = GetFilename(bytAllData)
```

```
        If strAllowedFiles <> "" Then
```

```
            If Not AllowedFile(strFilename) Then
```

```
                strError = "Filetype is not allowed."
```

```
                Upload = 3
```

```

Exit Function
End If
End If
bytdata = GetData(bytAllData)
Upload = 0
Else
    Upload = 1
    strError = "No data recieved."
End If
End Function

Public Function Save(byval bOverwrite)
    If strError <> "" Then
        Save = 4
        Exit Function
    End If
    If strPath <> "" Then
        If Mid(strPath,Len(strPath)-1,1) <> "\" Then strPath = strPath & "\"
        If strFilename <> "" Then
            If LenB(bytdata) > 1 Then
                If SaveBinaryData(bytdata,strPath &
strFilename,bOverwrite) Then
                    Save = 0
                Else
                    Save = 1
                End If
            Else
                Save = 2
                strError = "No data."
            End If
        Else
            Save = 3
            strError = "Not a valid filename specified."
        End If
    Else
        Save = 4
        strError = "No path specified."
    End If
End Function

Private Function AllowedFile(byval sFilename)'as boolean
    Dim arrAllowedFiles, intCount
    Dim strExtension
    If Len(sFilename) > 0 Then
        If InStr(sFilename,".") > 0 Then
            strExtension = Mid(sFilename,Len(sFilename) -
inStr(strReverse(sFilename),".")+2)
            arrAllowedFiles = Split(strAllowedFiles,",")

```

```

        AllowedFile = False
        For intCount = 0 To Ubound(arrAllowedFiles)
            If arrAllowedFiles(intCount) <> "" Then
                If Lcase(strExtension) =
Lcase(Trim(arrAllowedFiles(intCount))) Then
                    AllowedFile = True
                    Exit For
                End If
            End If
        Next
    Else
        AllowedFile = False
    End If
Else
    AllowedFile = False
End If
End Function

Private Function SaveBinaryData(byval bData, byval sFilename, byval bOverwrite)
'as boolean
    Dim objFs, objTextFile
    Dim intCount, strFile
    If LenB(bData) < 2 Then
        strError = "No data."
        SaveBinaryData = False
        Exit Function
    End If

    Set objFs = Server.CreateObject("scripting.filesystemobject")
    If Not objFs.FolderExists(strPath) Then
        strError = "Directory does not exists."
        SaveBinaryData = False
        Exit Function
    End If

    If Not bOverwrite And objFs.FileExists(sFilename) Then
        strError = "File already exists."
        SaveBinaryData = False
        Exit Function
    End If

    Set objTextFile = objFs.CreateTextFile(sFilename,True,False)

    For intCount = 1 To LenB(bData)
        objTextFile.Write Chr(AscB(MidB(bData,intCount,1)))
    Next

    objTextFile.Close

```

```

Set objTextFile = Nothing
Set objFs = Nothing
Session("file") = Null
SaveBinaryData = True
End Function

Private Function GetData(byval bFile) as bytearray
    Dim intStart, intEnd

    If LenB(bFile) < 1 Then
        GetData = ChrB(10)
        Exit Function
    End If
    intStart = InStrB(bFile, ChrB(13) & ChrB(10) & ChrB(13) & ChrB(10)) + 4
    intEnd = InStrB(bFile, ChrB(13) & ChrB(10) & ChrB(45) & ChrB(45) &
ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) &
ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) &
ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) &
ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45) & ChrB(45))
    If intStart > 0 Then
        If intStart < intEnd Then
            GetData = MidB(bFile, intStart, intEnd - intStart)
        Else
            GetData = ChrB(10)
        End If
    Else
        GetData = ChrB(10)
    End If
End Function

Private Function GetFilename(byval bFile) as string
    Dim bytFilename, bytChar, strFilename
    Dim intStart, intCount

    If LenB(bFile) < 1 Then
        GetFilename = ""
        Exit Function
    End If

    If LenB(bFile) > 0 Then
        If InStrB(bFile, ChrB(102) & ChrB(105) & ChrB(108) & ChrB(101) &
ChrB(110) & ChrB(97) & ChrB(109) & ChrB(101) & ChrB(61)) Then
            intStart = InStrB(bFile, ChrB(102) & ChrB(105) & ChrB(108) &
ChrB(101) & ChrB(110) & ChrB(97) & ChrB(109) & ChrB(101) & ChrB(61)) + 10
            For intCount = intStart To LenB(bFile)
                bytChar = MidB(bFile, intCount, 1)
                If bytChar = ChrB(34) Then
                    Exit For
                End If
            Next intCount
            strFilename = bytChar
        End If
    End If
End Function

```

```

                                End If
                                bytFilename = bytFilename & bytChar
                            Next
                        End If
                    End If
                    For intCount = 1 To LenB(bytFilename)
                        strFilename = strFilename &
Chr(AscB(MidB(bytFilename,intCount,1)))
                    Next
                    strFilename = Mid(strFilename,Len(strFilename) -
inStr(strReverse(strFilename),"\")+2)
                    GetFilename = strFilename
                End Function

Private Function GetContentType(byval bFile)
    Dim bytContentType, strContentType, bytChar
    Dim intStart, intCount

    If LenB(bFile) < 1 Then
        GetContentType = ""
        Exit Function
    End If

    If inStrB(bFile,ChrB(67) & ChrB(111) & ChrB(110) & ChrB(116) & ChrB(101)
& ChrB(110) & ChrB(116) & ChrB(45) & ChrB(84) & ChrB(121) & ChrB(112) &
ChrB(101) & ChrB(58)) > 0 Then
        intStart = inStrB(bFile,ChrB(67) & ChrB(111) & ChrB(110) &
ChrB(116) & ChrB(101) & ChrB(110) & ChrB(116) & ChrB(45) & ChrB(84) & ChrB(121)
& ChrB(112) & ChrB(101) & ChrB(58)) + 14
        For intCount = intStart To LenB(bFile)
            bytChar = MidB(bFile, intCount,1)
            If bytChar = ChrB(13) Then
                Exit For
            End If
            bytContentType = bytContentType & bytChar
        Next
    End If
    For intCount = 1 To LenB(bytContentType)
        strContentType = strContentType &
Chr(AscB(MidB(bytContentType,intCount,1)))
    Next
    GetContentType = strContentType
End Function
End Class

```

%>

حالا کافیسٲ در فایل اول در خط ۱۵ مسیر را برای ذخیره فایلها روی سرویس‌دهنده خود مشخص کنید :

```
;"Path = "D:\Inetpub\wwwroot\test.
```

مسیر تعیین شده باید حتماً موجود باشد.

ارسال ایمیل با کمک ASP

استفاده از آبجکت CDONTS که یکی از کامپوننتهای ویندوز سرور ۲۰۰۰ و ان تی می باشد به راحتی می توان در اسکریپتهای سمت سرور ایمیل ارسال کرد. طریقه استفاده به شکل زیر می باشد

```
<%  
Dim objNewMail  
Set objNewMail = Server.CreateObject("CDONTS.NewMail")  
%>
```

مقادیر قابل تنظیم برای این آبجکت به شرح جدول زیر می باشد.

From

یک مقدار رشته ای که آدرس پست الکترونیکی فرستنده را مشخص می کند
(به طور مثال Me@somewhere.com)

To

یک مقدار رشته ای که آدرس پست الکترونیکی گیرنده را مشخص می کند.
(به طور مثال, someone@somewhere.com)
آدرس گیرنده های مختلف با استفاده از علامت روبرو امکان پذیر میباشد ";"

Subject

عنوان پیام

Body

متن پیام

Cc

آدرس پست الکترونیکی شخصی که قرار است یک کپی از نامه را دریافت کند

Bcc

آدرس الکترونیکی شخصی که قرار است یک کپی از نامه را دریافت کند (بدون اطلاع دریافت کننده نامه)

Importance

یک مقدار عددی که نشان دهنده درجه اهمیت نامه می باشد
(برای مثال خیلی مهم , مهم , عادی)

BodyFormat

یک مقدار عددی که نشان دهنده فرمت اطلاعات متن نامه می باشد
ObjMail.MailFormat = 0 (Mime format)
ObjMail.BodyFormat = 1 (متن عادی)

MailFormat

یک مقدار عددی که نشان دهنده فرمت اطلاعات نامه می باشد
(ObjMail.MailFormat = 0 (Mime format)

ObjMail.MailFormat = 1 (متن عادي)

متودهاي اين آبجکت

AttachFile

این متود یک فایل را به نامه پیوست می کند

AttachURL

این متود آدرس فایل پیوست شده را تخصیص می دهد

Send

این متود نامه را ارسال می کند

بعد از ارسال ایمیل مقادیر در حافظه باقی می ماند و شما باید حافظه را خالی کنید. همچنین برای ارسال یک ایمیل جدید باید یک آبجکت جدید ایجاد کرده و از آن استفاده کنید. کد زیر نحوه استفاده از این آبجکت را در asp نشان می دهد.

```
<%
```

```
Option Explicit
```

```
Dim objNewMail
```

```
'ابتدا یک آبجکت از نوع cdonts ایجاد می کنیم
```

```
Set objNewMail = Server.CreateObject("CDONTS.NewMail")
```

```
'بعد از ایجاد آبجکت ما باید مقادیر مربوط به آن را تنظیم کنیم'
```

```
objNewMail.From = "webmaster@iran.com"
```

```
objNewMail.To = "test@yahoo.com"
```

```
objNewMail.Subject = "این یک ایمیل تستی است"
```

```
objNewMail.Body = "این متن نامه است"
```

```
objNewMail.Send
```

```
'بعد از ارسال ایمیل با استفاده از کد زیر فضای حافظه را خالی می کنیم'
```

```
Set objNewMail = Nothing
```

```
Response.Write "ایمیل ارسال شد"
```

```
%>
```

شي RESPONSE و استفاده از فايل هاي include

براي پاسخ گويي به مرورگر ، از متدهاي شيء Response استفاده مي شود . در واقع شيء Response راهي براي برقراري ارتباط با سرويس گيرنده است .

شيء Scripting Context :

اين شيء امکان دسترسى اشياء ActiveX خارجي به اشياء دروني ASP را فراهم مي کند. البته بهتر است به جاي اين شيء از شيء Object Context استفاده کنید . براي گرفتن شيء Object Context نيز بايد متد Object Context () را فراخواني کنید . شيء Scripting Context ، تمام اشياء ASP را درون خود جمع مي کند . هنگامى كه يك صفحه ASP كه شامل رفرنس هايي به اشياء ActiveX مي باشد اجراء ميشود ، ابتدا موتور ASP متد On Start Page هر يك از آن اشياء ActiveX را فراخواني کرده و شيء Scripting Context را به عنوان پارامتر به آن مي فرستد. اشياء ActiveX با استفاده از اين پارامتر مي توانند اشاره گر هايي به اشياء Server ، Application ، Request و Response به دست آورند. در اين جا ليستى از ويژگي هاي شيء Scripting Context را ملاحظه مي کنید :

- Server
- Application
- Session
- Request
- Response

هر يك از اين ۵ ويژگي يك اشاره گر (Pointer) به يكي از اشياء دروني ASP بر مي گردانند .
• توجه : Microsoft توصيه مي کند كه به جاي استفاده از شيء Scripting Context از متد get Object Context اگر چه امکان استفاده از شيء Scripting Context كماكان وجود دارد، ولي بايد بدانيد كه اين شيء از رده خارج شده و در برنامه هاي جديد خود نبايد از آن استفاده کنید .

شيء Object Context :

شيء Object Context به عنوان پل ارتباطي با MTS (Microsoft Transaction Server) تلقي مي شود. با MTS شما مي توانيد كاري كنيد كه اشياء ActiveX در تراكنش هايي كه صفحه ASP آغاز شده اند شركت كنند. همچنين با استفاده از شيء Object Context در هنگام نوشتن اشياء ActiveX مي توانيد به اشياء دروني ASP دسترسى پيدا كنيد .
آشنايي با ساختار يك برنامه ASP

از نظر موتور ASP ، يك برنامه کاربردي عبارت است از مجموعه اي از تمام فايل و زير دايركتوري هاي (Subdirectory) موجود در يك دايركتوري كه شامل فايل asa . Global باشد . بيشتر برنامه هاي کاربردي ASP شامل فايل هاي ASP و فايل هاي Include (با پسوند .inc) هستند كه هر دوي آنها مي توانند تركيبى از HTML ، كد و فايل هاي گرافيكى باشند. البته شما مي توانيد فايل هاي ASP را با فايل هاي HTML يا هر فايل ديگري كه براي سرويس دهنده قابل شناسايي باشد تركيب كنيد .

بالا ترين دايركتوري در اين ساختار شامل فايل asa . Global است . اين فايل دايركتوري ريشه (Root Directory) برنامه کاربردي را مشخص مي کند .

اولين باري كه يك كاربر يكي از فايل هاي ASP موجود در يكي از دايركتوري هاي برنامه را درخواست مي کند ، موتور ASP ساختار درختي درختي دايركتوري ها را به سمت بالا طي ميكند تا فايل asa . Global را پيدا كند و يا به ريشه وب (Root Web Directory) برسد . اين نكته از اهميت زيادي برخوردار است ، چون اگر شما چند برنامه کاربردي را درون هم قرار دهيد (كه به آنها برنامه هاي کاربردي تودرتو - Nested Applications - مي گويند) ، فايل asa . Global كه

برای هر درخواست کاربر اجرا می شود بستگی دارد به این که کاربر ابتدا کدام فایل را درخواست نماید . چون نمی توان اولین درخواست کاربر را کنترل نمود ، بهتر است برنامه های ASP را درون هم قرار ندهید ؛ مگر این که دلیل موجهی برای این کار داشته باشید .
ساختن یک برنامه کاربردی وب تحت IIS 4.0 به یک مرحله دیگر نیز دارد : شما باید به IIS بگویید که دایرکتوری مجازی (Virtual Directory) در برگیرنده فایل Global . asa ، دایرکتوری ریشه برنامه کاربردی است .

صرف نظر از این که کدام فایل درخواست شود، فایل Global . asa همواره اولین فایل خواهد بود که اجرا میشود. بنابراین اگر در نظر دارید کاربر برنامه کاربردی شما را با صفحه خاصی که مورد نظر شماست آغاز کند ، این فایل بهترین جایی است که میتوانید مرورگر را به آن صفحه هدایت کنید .

برنامه کاربردی که در شکل قبل ملاحظه فرمودید ، شامل چندین فایل ASP و دو زیردایرکتوری به نام های images و include می باشد. البته از نظر تعیین ساختار دایرکتوری های یک برنامه کاربردی هیچ محدودیتی وجود ندارد ؛ شما می توانید همه فایل ها را درون یک دایرکتوری قرار دهید . ولی در عمل ، بهتر است که فایل ها را بر اساس کارایی آنها دسته بندی کنید چون این کار ساختن و نگهداری برنامه را بسیار آسان تر می کند .

• توجه : دایرکتوری ریشه وب و دایرکتوری ریشه برنامه ASP لزوما نباید یکسان باشند. به عنوان مثال ،

می توانید برای چندبرنامه کاربردی ASP از یک فایل Global . asa استفاده کنید. در این شرایط می توانید فایل Global . asa را درون یک دایرکتوری قرار داده و سپس هر یک از زیر دایرکتوری های درون آن را به عنوان یک دایرکتوری مجازی تعریف کنید که بر اساس برنامه کاربردی مربوطه نامگذاری شده اند . ساختار دایرکتوری موجود در شکل زیر ، شامل ۴ برنامه کاربردی است : k۴۰۱، Paycheck، Retirement، Timesheet .

بالاترین دایرکتوری ، که HR Applications نام دارد، شامل فایل Global . asa است . هر زمان که یک کاربر به یکی از ۴ برنامه کاربردی موجود متصل شود ، موتور ASP ساختار درختی دایرکتوری هارا بالا می رود تا به دایرکتوری

HR Applications که در برگیرنده فایل Global . asa است برسد . یکی از دلایلی که می توان برای مرتب کردن برنامه ها به این شکل در نظر گرفت ، این است که ممکن است هر چهار برنامه از یک روش امنیتی استفاده کنند و یا ممکن است تمام برنامه ها از یک نوع روش برای برقراری ارتباط با پایگاه داده و یا از یک پایگاه داده مشترک استفاده کنند و شما بخواهید اطلاعات اولیه را در فایل Global . asa تنظیم کنید .

استفاده از فایل های Include :

برای جلوگیری از تکرار کد یا دستورات HTML در صفحات ASP ، می توانید از فایل های خارجی استفاده کرده و آنها را در صفحات ASP خود بگنجانید . یک فایل Include در یک فایل ASP از دستور INCLUDE استفاده می شود . دستور INCLUDE مشابه این است که محتویات فایل include مورد نظر را درون یک صفحه دیگر قرار دهید. در این جا مثالی از یک دستور INCLUDE را مشاهده می کنید :

```
< -- "INCLUDE = "C:\include\my Include .inc #--!>
```

این دستور سبب می شود که سرویس دهنده آن را با محتویات فایل مشخص شده جایگزین کند. دو نوع دستور INCLUDE موجود است : یکی INCLUDE FILE # و دیگری INCLUDE VIRTUAL # . نوع اول برای مشخص کردن فایل به یک آدرس فیزیکی نیاز دارد در حالی که نوع دوم به آدرس مجازی به فایل اشاره می کند . به هر حال از هر روشی که استفاده کنید، موتور ASP قبل از شروع پردازش دستورات ، آن را با محتویات فایل مورد نظر جایگزین می کند . بنابر این برای

گنجاندن يك فايل نمي توانيد از دستورات شرطي استفاده كنيد . به عنوان مثال ، كد زير به گونه‌اي كه انتظار مي رود عمل نخواهد كرد ؛ موتور ASP هر دو فايل را خواهد گنجاند :

```
< %  
if my Var = True than  
% >  
< ! -- # INCLUDE FILE = " C: \ include \ my Include . inc" -- >  
< %  
else  
%>  
< ! -- # INCLUDE FILE = " C: \ include \ your Include . inc" -- >  
< %  
end if  
% >
```

هنگامي كه موتور ASP اين فايل را تحليل مي كند ، دو دستور INCLUDE را با فايل هاي نامبرده جايگزين مي كند و اين كار را قبل از اجراي دستور if...end if انجام مي دهد . پس باز هم تأكيد مي كنم : شما با استفاده از كد نمي توانيد براي موتور ASP تعيين كنيد كه کدام فايل را بگنجاند و يا اين كه اصلا يك فايل را بگنجاند يا خير.

برنامه کاربردی تحت وب چیست ؟

يك برنامه کاربردی تحت وب بايك وب سایت تفاوت دارد . يك وب سایت شامل اطلاعات ثابتی است كه از پیش آماده شده و به صورت فایل های HTML ذخیره شده اند . در این حالت جهت حرکت اطلاعات از سرویس دهنده به

سرویس گیرنده می باشد . در واقع وب سایت اطلاعات را به کاربر ارائه می دهد . بین درخواست ها ، سرویس دهنده از عملکرد سرویس گیرنده اطلاعی ندارد . سرویس گیرنده می تواند از يك محل به محل دیگر ببرد و این کار روی وب سایت اولیه هیچ تأثیری ندارد ، چون هر يك از صفحات به صورت يك واحد كاملاً جدا می باشند . در واقع وب سایت شامل گروهی از فایل های HTML از هم جدا می باشد .

در مقابل ، يك برنامه کاربردی تحت وب ، اطلاعات را برای يك کاربر یا گروهی از کاربران به شکلی خاص فرمت کرده و ارائه می دهد . در این حالت ، جهت حرکت اطلاعات دو طرفه می باشد ؛ محتوایاتی كه توسط مرور گر

نمایش داده می شوند ، بر حسب اطلاعات ورودی کاربر یا هویت وی تعیین می گردند . بنابر این يك برنامه کاربردی تحت وب ، اطلاعات را به صورت دینامیکی ارائه می دهد نه به صورت فایل های HTML ثابت . در واقع وظیفه برنامه ، فراهم کردن اطلاعات بر حسب شرایط است كه این اطلاعات معمولاً از پایگاه داده (Database) بدست می آید . برنامه کاربردی نه تنها اطلاعات می دهد ، بلكه می تواند از کاربر اطلاعات بگیرد و بكارهای مختلف پاسخ مناسب دهد . به عبارت دیگر ، يك برنامه کاربردی ، عملکرد کاربر را از يك درخواست به

درخواست دیگر ردیابی می كند . بنابراین کاربر می تواند آن گونه كه در يك وب سایت به جستجو و گردش می پردازد ، به شكل دلخواه تمام صفحات يك برنامه کاربردی را بررسی و مشاهده كند . بلكه این برنامه کاربردی است كه صفحات مناسب را در شرایط مقتضی به طور دینامیکی تولید کرده و به کاربر ارائه می دهد .

برای نوشتن برنامه های کاربردی تحت وب از زبان های مختلفی می توان استفاده كرد ، مثل : Perl , Python , Java , C , C ++ , Visual Basic حتي Script های Unix .

در هر صورت مواردی كه در زیر به آنها اشاره می شود جزء ضروریات يك برنامه کاربردی هستند :

- ارتباط با يك پایگاه داده

- سرعت

- کار با رشته ها

- امنیت (Security)

- تراکنش ها (Transactions)

تراکنش چیست ؟

يك برنامه کاربردی تحت وب اغلب باید كارهای مختلفی انجام دهد كه هر يك از آنها برای تداوم اعتبار اطلاعات باید به طور كامل انجام شوند . هنگام كار با يك پایگاه داده ، معمولاً این كارها را بصورت يك تراکنش در نظر می گیریم . يك تراکنش قرار دادي است كه موفقیت یا عدم موفقیت مجموعه ای از عملیات را تضمین می كند كه در صورت عدم موفقیت ، اطلاعات باید دست نخورده باقی بمانند . به عنوان يك نمونه كلاسیك ، می توانیم يك حساب چك (Checking Account) را در نظر بگیریم .

هنگامی كه يك چك را به حساب خود می گذارید ، بانك مبلغ مشخص شده را از حساب مربوطه برداشته و به حساب شما واریز می كند . كل این عملیات در صورتی موفقیت آمیز خواهد بود كه برداشت و واریز پول به طور كامل انجام شود . در این حالت عمل دو گانه برداشت از يك حساب و واریز به يك حساب دیگر را يك تراکنش می گوئیم .

تفاوت میان برنامه های IIS و DHTML

برنامه های IIS ، برنامه هایی هستند كه بر روی سرویس دهنده و تحت Microsoft Internet Information Server اجرا می شوند . در مقابل ، برنامه های DHTML برنامه Microsoft Internet Explorer استفاده می كنند و به این ترتیب به ورودی های کاربر پاسخ سریع می دهند .

این برنامه ها برای نوشتن بازی ها و شبیه سازی محیط و گرفتن اطلاعات از کاربر مناسب هستند . متأسفانه این نوع برنامه ها به نسخه چهارم IE یا نسخه های بالاتر آن نیاز دارند . با توجه به این که در این کتاب به تفصیل در مورد ASP و مسائل حاشیه ای آن بحث خواهیم کرد ، بد نیست کمی بیشتر در مورد برنامه های IIS توضیح دهیم . چون هدف از یادگیری ASP ، نوشتن برنامه های IIS می باشد که بر روی سرویس دهنده اجرا می شوند .

برنامه های IIS با چندین مرورگر مختلف و نسخه های متفاوت آنها سازگاری دارند . همچنین در محیط های مختلف Mak یا Unix نیز اجراء می شوند . بنابر این این نوع برنامه ها برای نوشتن برنامه های تجاری که مخاطبین زیادی دارند مناسب هستند ؛ چون دلیلی ندارد همه کاربران از ویندوز استفاده کرده و یا نسخه IE را در اختیار داشته باشند.

با توجه به این که برنامه های IIS بر روی سرویس دهنده اجرا می شوند ، برای به روز در آوردن آنها تنها کافی است کد موجود بر روی سرویس دهنده را تغییر داد. این نوع برنامه ها از هر محلی قابل دسترسی می باشند . هنگامی که یک برنامه IIS بر روی سرویس دهنده نصب شده و به اجرا در آمد ، هر کامپیوتری که بتواند به آن سرویس دهنده متصل شود می تواند از آن برنامه استفاده کند .

شما می توانید برای شناسایی کاربران ، به آنها نام و کلمه عبور اختصاص دهید و هنگام برقراری ارتباط با برنامه کاربردی ، نام و کلمه عبور را در خواست کنید . با توجه به این که تمام اطلاعات بر روی یک کامپیوتر ذخیره می شوند ، هنگامی که کاربر برنامه کاربردی را ترک کند ، هیچ نوع اطلاعاتی را از دست نمی دهد .

در قسمت بعد ، در مورد چگونگی کارکرد برنامه های IIS بحث خواهیم کرد .
چگونگی در خواست فایل ها توسط مرورگر :

هنگامی که شما یک آدرس URL در فیلد آدرس مرورگر خود تایپ می کنید ، چند چیز اتفاق می افتد . مرورگر ، آدرس وارد شده را تحلیل کرده و پیغامی به سرویس دهنده نام (Name Server) می فرستد تا نام مشخص شده

(به عنوان مثال ، Microsoft . com) را به یک آدرس (Internet Protocol) IP – (به عنوان مثال ، ۲۰۷,۸۴,۲۵,۳۲) ترجمه کند . سپس مرورگر با استفاده از آن آدرس IP به سرویس دهنده مربوطه متصل شده و فایل را درخواست می کند . سرویس دهنده فایل را می خواند و محتویات آن را به مرورگر بر می گرداند . در این مرحله مرورگر ، دستورات HTML آن فایل را تحلیل و فرمت فایل را تعیین می کند . بیشتر فایل های HTML درون خود به تصاویر گرافیکی اشاره می کنند. این اشاره ها به صورت رفرنس (Reference) بوده که هریک از آنها نیز بایک آدرس URL مشخص می شوند .

بنابراین کل این عملیات برای تکه تکه تصاویر و یا فایل های دیگری که با رفرنس به آنها اشاره شده است تکرار می شود.

به این ترتیب پروسه نمایش یک فایل HTML شامل تعدادی تراکنش کوچک بین سرویس گیرنده (یعنی مرورگر) و سرویس دهنده وب می باشد .

برنامه های IIS نیز تقریباً به همین صورت عمل می کنند. ابتدا یک مرورگر یک صفحه را از سرویس دهنده درخواست می کند . این درخواست همیشه برای یک فایل خاص می باشد . پاسخ سرویس دهنده به نوع فایل درخواستی بستگی دارد. اگر فایل مورد نظر از نوع HTML (یا پسوند .html یا .htm) باشد ، سرویس دهنده به سادگی محتویات فایل را می خواند و محتویات آنرا به صورت یک رشته در آورده و براساس قوانین URL کد گذاری می کند و سپس رشته حاصله را به مرورگر برمی گرداند . کل این پروسه ، از درخواست تا پاسخ ، یک تراکنش مابین سرویس گیرنده و سرویس دهنده تلقی می شود . سرویس گیرنده همواره تراکنش را شروع می کند و سپس تا زمان رسیدن پاسخ از سوی سرویس دهنده منتظر می ماند که در این زمان تراکنش کامل شده و خاتمه می یابد .

درخواست فایل از سرویس دهنده وب مشابه هنگامی است که در Windows Explorer روی یک فایل شبکه ای (فایلی که بر روی کامپیوتر سرویس دهنده قرار دارد) دوبار – کلیک می کنید ؛ البته با دو تفاوت :

- سرویس دهنده وب هیچگاه اجازه نمی دهد که برنامه محلی شما (یعنی مرورگر) فایل درخواستی را باز کرده یا در آن چیزی بنویسد؛ در عوض خودش فایل را باز می کند و محتویات آن را برمیگرداند.

- ارتباط مابین سرویس گیرنده و سرویس دهنده گذرا (Transient) است. برای برقراری ارتباط با سرویس دهنده وب نیازی به مشخص کردن نام درایوی که فایل بر روی آن قرار دارد نیست. پس از این که سرویس دهنده وب به درخواست شما پاسخ داد، ارتباط را قطع می کند. هنگامی که تراکنش کامل شد، سرویس دهنده وب، سرویس گیرنده را به طور کلی از یاد می برد. اگر بلا فاصله کلید Refresh (موجود بر روی نوار ابزار IE) را کلیک کنید، سرویس دهنده وب، آن تراکنش را تکرار می کند و اصلاً به خاطر نمی آورد که ۵ ثانیه قبل این فایل توسط شما درخواست شده است.

اگر ارتباط براساس پروتکل استاندارد HTTP برقرار شود، بیشتر فایل ها به چندین تراکنش نیاز دارند، یکی برای فایل Html اصلی و بقیه برای یکایک فایل های گرافیکی که در فایل اصلی به آنها اشاره شده است. بنابراین برای نمایش يك فایل که شامل ۵ تصویر گرافیکی است، مرورگر درخواست جداگانه به سرویس دهنده وب می فرستد.

فرم ها در ASP

مثالها

يك فرم با متد "get"

چگونه با استفاده از دستور Request.QueryString به کاربر عكس العمل نشان دهيد.

يك فرم با متد "post"

چگونه با استفاده از دستور Request.Form به کاربر عكس العمل نشان دهيد.

يك فرم با Radio Button ها

چگونه با استفاده از دستور Request.Form و از طريق Radio Button ها به کاربر عكس العمل نشان دهيد.

ورودی کاربر

آبجكت Request می تواند برای گرفتن اطلاعات از کاربر از فرمها استفاده گردد.

```
<form method="get" action="simpleform.asp">  
First Name: <input type="text" name="fname">  
<br />  
Last Name: <input type="text" name="lname">  
<br /><br />  
<input type="submit" value="Submit">  
</form>
```

ورودی های کاربر می توانند از دو طریق استفاده گردد: با Request.QueryString یا Request.Form

Request.QueryString

دستور Request.QueryString برای جمع کردن مقادیر موجود در يك فرم با متد "get" استفاده می شود. اطلاعات فرستاده شده از فرمی با متد "get" برای هر کسی قابل مشاهده است (این اطلاعات در قسمت آدرس Browser نشان داده می شود) و در مورد مقدار اطلاعات محدودیت دارد.

اگر کاربر در فرمی با متد "get" مقدار "Bill" و "Gates" را وارد کنید، URLی که به سرور فرستاده می شود شبیه به این خواهد بود:

<http://www.irandevlopers.com/simpleform.asp?fname=Bill&lname=Gates>

فرض کنید فایل "simpleform.asp" حاوی اسکریپت زیر است:

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


</body>

Browser خط زیر در قسمت Body سند نشان خواهد داد:

Welcome Bill Gates

Request.Form

دستور Request.Form برای جمع کردن اطلاعات از يك فرم با متد "post" استفاده می‌شود. اطلاعات فرستاده شده از فرمی با متد POST برای بقیه غیرقابل مشاهده است و هیچ محدودیتی در مقدار اطلاعات فرستاده شده ندارد.

اگر کاربر در فرمی با متد "post" کلمه "Bill" و "Gates" را تایپ کند، URLی که به سرور فرستاده می‌شود شبیه به این خواهد بود:

<http://www.w3schools.com/simpleform.asp>

فرض کنید که فایل "simpleform.asp" حاوی اسکریپت زیر باشد:

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

Browser خط زیر در قسمت Body سند نشان خواهد داد:

Welcome Bill Gates

تصدیق فرم یا تست معتبر بودن (درست بودن) مقادیر ورودی

ورودی کاربر باید در Browser در هر جایی که امکان داشت (با استفاده از اسکریپت‌های client) تصدیق گردد. تصدیق Browser سریعتر است و زمان load شدن سرور را نیز کاهش می‌دهد.

در صورتی که مقادیر ورودی کاربر وارد Database می‌شود لازم است که معتبرسازی یا تصدیق ورودی‌های کاربر بر روی سرور انجام شود. يك راه خوب برای تصدیق ورودی‌های کاربر، به جای فرستادن اطلاعات به يك صفحه دیگر، پست کردن آنها به خود فایل است. در آن صورت کاربر می‌تواند خطاها را در خود صفحه‌ای که فرم در آن قرار دارد مشاهده نماید. این پیدا کردن خطا را برای وی ساده‌تر می‌کند.

طرز ساخت کوکی ها

يك Cookie اغلب برای شناسائی کاربر استفاده می‌شود.

مثالها

Cookie خوش‌آمدگویی
چگونه يك Cookie خوش‌آمدگویی درست کنیم.

يك Cookie چیست؟

يك Cookie اغلب برای شناسندن کاربر استفاده می‌شود. يك Cookie فایل كوچك است كه سرور آن را در كامپیوتر کاربر جاسازی می‌کند. هر بار كه همان كامپیوتر يك صفحه را درخواست می‌کند، Cookie نیز فرستاده می‌شود. با ASP شما می‌توانید هم مقادیر Cookie را بسازید و هم آنها را استخراج نمایید.

چگونه يك Cookie بسازید؟

Response.Cookie برای ساختن Cookie استفاده می‌شود.

توجه: Response.Cookie باید قبل از تگ درج شود.

در مثال زیر، ما يك Cookie درست خواهیم كردم و اسم آن را "firstname" می‌گذاریم و مقدار "Alex" را به آن می‌دهیم:

همچنین امکان دارد كه خصوصياتی را مثل تاریخ انقضای cookie را به آن بدهید:

چگونه مقدار cookie ها را استخراج کنیم

دستور "Request.Cookies" برای استخراج اطلاعات از cookie ها استفاده می‌شود.

در مثال زیر، ما مقدار cookie ی با نام "firstname" را استخراج کرده و در صفحه نشان دادیم:

خروجی:

Firstname=Alex

يك Cookie با کلید

اگر يك cookie حاوی مجموعه‌ای از مقادیر باشد، می‌گوییم آن cookie دارای کلیدهایی است.

در مثال زیر، ما يك مجموعه cookie با نام "user" درست می‌کنیم. cookie به نام "user" دارای کلیدهایی است كه حاوی اطلاعاتی راجع به کاربر است.

```
<%  
Response.Cookies("user")("firstname")="John"  
Response.Cookies("user")("lastname")="Smith"  
Response.Cookies("user")("country")="Norway"  
Response.Cookies("user")("age")="25"  
>%
```

خواندن همه Cookieها

به کد زیر نگاه کنید:

```
%>
Response.Cookies("firstname")="Alex"
Response.Cookies("user")("firstname")="John"
Response.Cookies("user")("lastname")="Smith"
Response.Cookies("user")("country")="Norway"
Response.Cookies("user")("age")="25"
<%
```

فرض کنیم که سرور شما همه cookieهای بالا را به کاربر فرستاده است.

حالا ما می‌خواهیم تمام cookieها را که به کاربر فرستاده شده است را بخوانیم. مثال زیر نشان می‌دهد که چگونه این کار را انجام دهیم (توجه داشته باشید که در مثال پایین با استفاده از HasKeys چک می‌کند که آیا cookie کلید دارد یا نه؟)

```
<html>
<body>
<%
dim x,y
for each x in Request.Cookies
  response.write("<p>")
  if Request.Cookies(x).HasKeys then
    for each y in Request.Cookies(x)
      response.write(x & ":" & y & "=" & Request.Cookies(x)(y))
      response.write("<br />")
    next
  else
    Response.Write(x & "=" & Request.Cookies(x) & "<br />")
  end if
  response.write "</p>"
next
%>
</body>
</html>
```

خروجی:

```
firstname=Alex user:firstname=John user:lastname=Smith user: country=Norway user:
age=25
```

اگر Browser دارای پشتیبانی Cookie نباشد چه اتفاقی می‌افتد؟

اگر برنامه شما با Browserی سرو کار داشته باشد که cookie را پشتیبانی نمی‌کند، باید از روش‌های دیگری برای انتقال اطلاعات از يك صفحه به صفحات دیگر در برنامه خود استفاده کنید. دو راه برای انجام این کار هست:

۱. اضافه کردن پارامترها به URL

شما می‌توانید پارامترها را به URL اضافه کنید:

```
<a href="welcome.asp?fname=John&lname=Smith">  
Go to Welcome Page</a>
```

و مقادیر موجود در "welcome.asp" را همچون مثال زیر استخراج کنید:

۲. استفاده از form

شما می‌توانید از فرم‌ها استفاده کنید. وقتی که کاربر روی دکمه Submit کلیک کند فرم ورودی‌های کاربر را به "welcome.asp" رد می‌کند:

```
<"form method="post" action="welcome.asp">  
<""=First Name: <input type="text" name="fname" value  
<""=Last Name: <input type="text" name="lname" value  
<"input type="submit" value="Submit">  
<form/>
```

مقادیر موجود در "welcome.asp" را همچون این مثال استخراج کنید:

```
<form method="post" action="welcome.asp">  
First Name: <input type="text" name="fname" value="">  
Last Name: <input type="text" name="lname" value="">  
<input type="submit" value="Submit">  
</form>
```

آبجکت Response در ASP

شما نمی توانید سورس کد های ASP را بوسیله انتخاب گزینه View Source در Browser مشاهده کنید، شما فقط خروجی فایل ASP را مشاهده می کنید که HTML ساده است. زیرا که اسکریپت ها قبل از این که نتایج به Browser فرستاده شود بر روی سرور اجرا می گردد.

قواعد اساسی نحو

یک فایل ASP به طور عادی دقیقا همانند HTML حاوی Tag های HTML است. اگر چه، یک فایل ASP می تواند همچنین حاوی Script ها نیز باشد. این اسکریپت ها به وسیله علامت های %< و %> احاطه شده اند. اسکریپت های سرور بر روی سرور اجرا می شود و می تواند حاوی هر گونه expressions، دستورات (Statements)، توابع، یا عملگرهایی باشد.

آبجکت Response

متد Write مربوط به آبجکت Response برای فرستادن محتویات به Browser استفاده می شود. به عنوان مثال، دستورات زیر برای فرستادن متن "سلام دنیا!" به Browser استفاده می شود.

```
<%  
response.write("سلام دنیا!")  
%>
```

VBScript

این امکان وجود دارد که از زبان های مختلف اسکریپت نویسی در فایل های ASP استفاده کنید. اگرچه زبان پیش فرض، VBScript است.

```
<html>  
<body>  
<%  
response.write("سلام دنیا!") %>  
</body>  
</html>
```

مثال بالا "سلام دنیا!" را در قسمت body متن می نویسد.

JavaScript

برای این که JavaScript را به عنوان زبان پیش فرض برای یک صفحه خاص مشخص کنید می بایست از راهنماهای زبانی در ابتدای صفحه خود استفاده کنید:

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

نکته: JavaScript برعکس VBScript به بزرگ یا کوچک بودن حروف حساس است. شما مجبور خواهید بود که وقتی زبان احتیاج دارد، کد ASP خود را با حروف بزرگ یا کوچک بنویسید.

سایر زبان های اسکریپ نویسی

ASP همراه با VBScript و JScript (پیاده سازی میکروسافت از JavaScript) عرضه می شود. اگر شما می خواهید که با زبان دیگری همچون PERL - REXX یا Python کد بنویسید، مجبور خواهید بود که موتور Script مربوط به آنها را نصب کنید. مهم: به خاطر این که اسکریپت ها بر روی سرور اجرا می شوند، به هیچ وجه لازم نیست Browser ی که فایل ASP را نشان می دهد از Scripting پشتیبانی کند!

راه اندازی ASP

شما می توانید ASP را بدون نیاز به یک سرور خارجی اجرا کنید. برای انجام این کار شما باید Microsoft's Personal Web Server (PWS) یا Internet Information Server (IIS) را در رایانه خود نصب کنید.

چگونه ASP را در رایانه خود اجرا کنید؟

شما می توانید ASP را بدون سرور خارجی در رایانه شخصی خود اجرا کنید. برای انجام این کار باید Microsoft's (Personal Web Server (PWS یا Internet Information Server (IIS) را در رایانه خود نصب کنید.

اگر در مورد اجرای ASP جدی هستید، باید حداقل ویندوز ۹۸، نسخه دوم (Second Edition) داشته باشید.

اگر در مورد اجرای ASP بسیار جدی هستید، باید از ویندوز ۲۰۰۰ استفاده کنید.

چگونه PWS را در ویندوز ۹۵ نصب کنید و ASP اجرا کنید؟

Personal Web Server (PWS) همراه ویندوز ۹۵ نیست!!

برای اجرای ASP بر روی ویندوز ۹۵، شما باید "Windows NT 4.0 Option Pack" را از مایکروسافت داون لود کنید.

داون لود "Windows NT 4.0 Option Pack"

چگونه PWS را در ویندوز NT نصب کنید و ASP اجرا کنید؟

Personal Web Server (PWS) همراه ویندوز NT نیست!!

برای اجرای ASP بر روی ویندوز ۹۵، شما باید "Windows NT 4.0 Option Pack" را از مایکروسافت داون لود کنید.

داون لود "Windows NT 4.0 Option Pack"

چگونه PWS را در ویندوز ۹۸ نصب کنید و ASP اجرا کنید؟

۱. شاخه Add-ons را در سی دی ویندوز ۹۸ خود باز کنید، شاخه PWS را پیدا کرده و فایل setup.exe را اجرا کنید.

۲. یک شاخه Inetpub در هارد شما ایجاد خواهد شد. آن را باز کنید و شاخه wwwroot را پیدا کنید.

۳. یک شاخه جدید درست کنید و نام آن را چیزی شبیه به "MyWeb" بگذارید.

۴. از یک ویرایشگر متن استفاده کنید و کدهای ASP را بنویسید، فایل را به نام "test1.asp" را در شاخه "MyWeb" ضبط کنید.

۵. مطمئن شوید که وب سرور شما اجرا است - برنامه نصب یک آیکن جدید در Task Bar شما اضافه کرده است (این نشانه PWS است). بر روی آن کلیک کنید و دکمه Start را در پنجره ای که آمده بزنید.

۶. Browser خود را باز کنید و تایپ کنید "http://localhost/MyWeb/test1.asp"، تا اولین صفحه ASP خود را ببینید.

چگونه PWS را در ویندوز Me نصب کنید و ASP اجرا کنید؟

Personal Web Server (PWS) در ویندوز Me وجود ندارد !!
مقاله مربوطه را در سایت مایکروسافت بخوانید

چگونه IIS را در ویندوز ۲۰۰۰ نصب کنید و ASP اجرا کنید؟

۱. از دکمه Start به Settings و سپس Control Panel بروید.
۲. در پنجره Control Panel آیکن Add/Remove Programs را انتخاب کنید.
۳. در پنجره Add/Remove Programs گزینه Add/Removes Windows Components را انتخاب کنید.
۴. در پنجره جدید گزینه Internet Information Services را انتخاب کرده، OK را بزنید.
۵. یک شاخه Inetpub در هارد شما ایجاد خواهد شد. آن را باز کنید و شاخه wwwroot را پیدا کنید.
۶. یک شاخه جدید درست کنید و نام آن را چیزی شبیه به "MyWeb" بگذارید.
۷. از یک ویرایشگر متن استفاده کنید و کدهای ASP را بنویسید، فایل را به نام "test1.asp" در شاخه "MyWeb" ضبط کنید.
۸. مطمئن شوید که وب سرور شما اجرا است - برنامه نصب یک آیکن جدید در Task Bar شما اضافه کرده است (این نشانه IIS است). بر روی آن کلیک کنید و دکمه Start را در پنجره ای که آمده بزنید.
۹. Browser خود را باز کنید و تایپ کنید "http://localhost/MyWeb/test1.asp"، تا اولین صفحه ASP خود را ببینید.

چگونه IIS را در ویندوز XP نصب کنید و ASP اجرا کنید؟

نکته: شما نمی توانید ASP را روی ویندوز XP نسخه خانگی (Home Edition) اجرا کنید.

۱. از دکمه Start به Settings و سپس Control Panel بروید.
۲. در پنجره Control Panel آیکن Add/Remove Programs را انتخاب کنید.
۳. در پنجره Add/Remove Programs گزینه Add/Removes Windows Components را انتخاب کنید.
۴. در پنجره جدید گزینه Internet Information Services را انتخاب کرده، OK را بزنید.
۵. یک شاخه Inetpub در هارد شما ایجاد خواهد شد. آن را باز کنید و شاخه wwwroot را پیدا کنید.
۶. یک شاخه جدید درست کنید و نام آن را چیزی شبیه به "MyWeb" بگذارید.

۷. از یک ویرایشگر متن استفاده کنید و کدهای ASP را بنویسید، فایل را به نام "test1.asp" را در شاخه "MyWeb" ضبط کنید.

۸. مطمئن شوید که وب سرور شما اجرا است - برنامه نصب یک آیکن جدید در Task Bar شما اضافه کرده است (این نشانه IIS است). بر روی آن کلیک کنید و دکمه Start را در پنجره ای که آمده بزنید.

۹. Browser خود را باز کنید و تایپ کنید "http://localhost/MyWeb/test1.asp"، تا اولین صفحه ASP خود را ببینید.

شروع کار با ASP

يك فایل ASP می تواند حاوی متون، Tag های HTML و اسکریپت ها باشد. اسکریپت ها در يك فایل ASP بر روی سرور اجرا می شوند.

به عنوان پیش نیاز چه چیزی را باید بدانید؟
قبل از شروع، باید با مسائل زیر را تا حدودی آشنا باشید

WWW و HTML و مسائل اولیه ساخت صفحات Web
يك زبان اسکریپت نویسی همچون JavaScript یا VBScript

ASP چیست؟

ASP حروف اختصاری Active Server Pages است.
ASP برنامه ای است که درون IIS اجرا می شود.
IIS حروف اختصاری Internet Information Services است.
IIS به عنوان يك مولفه رایگان با ویندوز ۲۰۰۰ عرضه می شود.
IIS همچنین به عنوان جزئی از Windows NT 4.0 Option Pack است.
Option Pack می تواند از سایت مایکروسافت داون لود شود.
PWS نسخه کوچکتر - ولی کاملاً عمل کننده - از IIS است.
PWS را می توانید در سی دی ویندوز ۹۵ یا ۹۸ خود پیدا کنید.

سازگاری ASP

ASP تکنولوژی مایکروسافت است.
برای اجرای IIS باید ویندوز NT نسخه ۴,۰ یا بالاتر را داشته باشید.
برای اجرای PWS باید ویندوز ۹۵ یا بالاتر را داشته باشید.
ChiliASP تکنولوژی است که ASP را بدون نیاز سیستم عامل ویندوز اجرا می کند.
InstantASP تکنولوژی دیگری است که ASP را بدون نیاز به سیستم عامل ویندوز اجرا می کند.

يك فایل ASP چیست؟

يك فایل ASP تقریباً شبیه به يك فایل HTML است.
يك فایل ASP می تواند حاوی متن، XML، HTML، و اسکریپت باشد.
اسکریپت های درون يك فایل ASP بر روی Server اجرا می شود.
يك فایل ASP پسوند ".asp" دارد.

فرق ASP با HTML چیست؟

وقتی که Browser (کاوشگر اینترنت) درخواست يك فایل HTML را بکند، Server آن فایل را بر می گرداند.
وقتی که Browser درخواست يك فایل ASP را بکند، IIS درخواست مربوطه را به موتور (هسته ای که وظیفه اصلی را بازی می کند) ASP ارجاع می دهد. موتور ASP فایل ASP را خط به خط می خواند، و اسکریپت های درون فایل را اجرا می کند. در نهایت فایل ASP به عنوان يك فایل HTML ساده به Browser برگردانده می شود.

ASP چه کاری می تواند برای شما انجام دهد؟

به طور دینامیکی محتویات يك Web page را ویرایش یا اضافه کند.
به درخواست های کاربر که از فرم های HTML ارسال شده اند پاسخ دهد.
به هر گونه اطلاعات با بانک اطلاعاتی دسترسی پیدا کرده و نتایج را به Browser برگرداند.
سفارشی نمودن (Customize) يك Web page تا برای کاربران مختلف مفیدتر باشد.
مزایای استفاده از ASP به جای CGI و Perl سرعت و سادگی است.
فراهم کردن امنیت بیشتر به گونه ای که کد ASP شما از طریق Browser قابل مشاهده نیست.
از آن جایی که فایل های ASP به شکل HTML برگردانده می شوند، میتوانند بوسیله هر Browser مشاهده گردند.
برنامه نویسی درست ASP می تواند ترافیک شبکه را به حداقل برساند.



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

مجموعه مقالات ای اس پی دات نت

ASP.net

Namespace چیست؟

یک نکته مهم که در زمان استفاده از .NET Framework باید به آن توجه داشت آن است که فضا نام (namespace یا نامکده) ها در ساختمان برنامه کاربردی قرار دارند. فضا نام یک طرح نامگذاری منطقی برای گروه بندی کلاس های مرتبط است. این طرح مانع از آن می شود تا کلاس هایی که برای متدها و خصوصیات از یک شناسه یکسان استفاده می کنند تداخل داشته باشند.

مثلا .NET Framework برای گروه بندی تایپ ها به مقوله های منطقی عمل کرد، از قبیل چارچوب برنامه کاربردی ASP.NET، از یک طرح نامگذاری سلسله مراتبی استفاده می کند. ابزارهای طراحی از فضا نام ها با هدف تسهیل مرور و ارجاع تایپ ها در برنامه بهره برداری می کنند. مثلا فرض کنید در حال نوشتن کد زیر هستید:

```
Public Class NewClass
    [Procedures and Functions]
End Class
```

```
Public Class NewClass
    [Procedures and Functions]
End Class
```

این کد به خطا منجر می شود چون کامپایلر راهی برای تشخیص کلاس ها از یکدیگر ندارد. برای غلبه بر این مشکل می توان از یک فضا نام استفاده کرد که اجازه می دهد دو کلاس هم نام در صفحه با هم وجود داشته باشند. قطعه برنامه زیر تعریف این دو کلاس در فضا نام های منحصر بفرد را نشان می دهد:

```
Namespace One
    Public Class NewClass
        [Procedures and Functions]
    End Class
End Namespace
```

```
Namespace Two
    Public Class NewClass
        [Procedures and Functions]
    End Class
End Namespace
```

در این کد برخوردی بین دو کلاس با نام NewClass وجود ندارد چون هر کدام در یک فضا نام جداگانه قرار داده شده است. کلاس اول را می توان با استفاده از ترکیب One.NewClass صدا زد، حال آنکه کلاس دوم را می توان با استفاده از ترکیب Two.NewClass صدا زد.

شما می توانید در فضا نام های خود از یک ساختار سلسله مراتبی استفاده کنید. قرار دادن اشیاء مشابه تحت زیرعنوانها در یک فضا نام مشترک تشخیص هدف فضا نام را آسانتر می کند، و در عین حال باعث می شود برنامه به مراتب شی گرا تر شود.

برای توضیح فضا نام می توان ساختار فایل و دایرکتوری (کشو، فولدر) در یک کامپیوتر را در نظر گرفت. در این مثال کلاس ها به مثابه فایل ها و فضا نام ها مانند دایرکتوری ها هستند. بدیهی

است همانگونه که می توانیم دایرکتوریهای تو در تو داشته باشیم، فضا نام ها هم در حالیکه کلاس ها را در خود جای داده اند می توانند بصورت تو در تو باشند.

فضا نام ها در ساخت برنامه های کاربردی ASP.NET نقش مهمی ایفا می کنند. خوشبختانه لازم نیست برای همه اشیا یی که می توانند به وسیله صفحات ASP.NET مورد استفاده قرار بگیرند سیستم طبقه بندی فضا نام ایجاد کنید. مایکروسافت این مساله را برای شما حل کرده است. دو فضا نام ریشه، و فضا نامهای فرزند آنها را می توان وارد صفحات ASP.NET خود کرد. اولی System نامیده می شود، و دومی Microsoft نام دارد. این فضا نامها با جزئیات بیشتر در ادامه مورد بحث قرار گرفته اند.

فضا نام System

فضا نام System فضا نام اصلی برای ساخت ASP.NET و همه برنامه های کاربردی دیگر مبتنی بر .NET Framework است. هر چیزی که در برنامه کاربردی شما قابل انجام باشد از طریق فضا نام System کنترل می شود. به عنوان مثال کنترل آرایه، عملیات ریاضی، و تبدیل نوع داده ها از طریق فضا نام System و فضا نامهای فرزند آن اداره می شوند. ۹ فضا نام پیش فرض (فضا نام System و ۸ فرزند آن) وجود دارند که به صورت خودکار به صفحات ASP.NET اضافه می شوند:

- System
- System.ComponentModel.Design
- System.Data
- System.Drawing
- System.Web.SessionState
- System.Web
- System.Web.UI
- System.Web.UI.WebControls
- System.Web.UI.HtmlControls

هشت فضا نام (بجز فضا نام System) در زمان ساخت Visual Studio.NET یا VS.NET بطور خودکار به صفحات ASP.NET وارد می شوند. این فضا نام ها در زیر به اختصار شرح داده شده اند.

• System.ComponentModel.Design: دربرگیرنده کلاس هایی است که می توان از آنها برای طراحی پشتیبانی سفارشی اجزا و زمان طراحی و دسترسی به سرویس های تامین شده توسط معماری .NET Framework استفاده کرد.

• System.Data: امکان دسترسی به کلاس ها و رابطهایی را فراهم می کند که معماری ADO.NET را برای دسترسی به داده های عمومی تشکیل می دهند.

• System.Drawing: دربرگیرنده کلاس ها و رابطهایی است که عملکرد گرافیکی اولیه را تامین می کنند. فضا نام System.Drawing نیز از طریق فضا نام System.Drawing.Drawing2D و System.Drawing.Imaging عملکرد پیشرفته تری فراهم می کند.

• System.Web: کلاس ها و رابطهایی تامین می کند که ارتباط مرورگر/سرویس دهنده را امکان پذیر می کنند. این فضا نام دربرگیرنده کلاس HttpRequest (فراهم کننده اطلاعات وسیعی درباره درخواست HTTP جاری)، کلاس HttpResponse (فراهم آورنده امکان دسترسی به فرایندها و یوتیلیتی های سمت سرویس دهنده) است.

- **System.Web.SessionState**: فراهم کننده کلاس ها و متدهایی برای مدیریت وضعیت جلسات کاری می باشد.

- **System.Web.UI**: فراهم کننده کلاس ها و رابطهای برای رابط واسط کاربر برنامه کاربردی ASP.NET است که موجب می شوند برنامه کاربردی با سطوح مختلف صفحه، ارتباط برقرار کند. کلاس اصلی این فضا نام، کلاس Page می باشد که دربرگیرنده همه خصوصیتها، متدها، و سازنده های صفحه است. اشیاء اصلی Active Server Page زیر خصوصیتهایی در کلاس Page هستند: Session و Server، Request، Response، Application.

- **System.Web.UI.HtmlControls**: کلاس هایی برای عناصر HTML استاندارد، شامل فرم ها، کنترل های ورودی، آنکور، جداول، قسمتهای متنی، و غیره فراهم می کند. این کنترلها همانند تگهای عادی HTML هستند با این تفاوت که دارای دو صفت "runat="server" و "id="controlname" می باشند.

- **System.Web.UI.HtmlControls**: برای کنترلهای سرویس دهنده ای که شبیه کنترلهای HTML هستند ولی انعطاف پذیری بیشتر و عملکرد پیچیده تری دارند کلاس هایی را تامین می کند.

برخی فضا نام های مهم و پرکاربرد دیگر به شرح زیر می باشند.

- **System.IO**: دربرگیرنده رابطها و کلاس هایی است که امکان خواندن و نوشتن همگام و غیرهمگام فایل ها و جریانهای داده را فراهم می کنند.

- **System.Data.OleDb**: امکان دسترسی به کلاس ها و رابطهای مخصوص دسترسی به یک منبع داده از طریق ADO را فراهم می کند.

- **System.Data.SqlClient**: امکان دسترسی به کلاس ها و رابطهای مخصوص دسترسی به داده های خاص Microsoft SQL Server از طریق ADO را فراهم می کند.

- **System.Web.Security**: امکان دسترسی به کلاسها و رابطهای مخصوص امنیت برنامه کاربردی ASP.NET را فراهم می کند. دستیابی به رمزنگاری، مجوزها، و تنظیمات خط مشی برنامه کاربردی در این فضا نام قرار می گیرند.

- **System.XML**: امکان دسترسی به کلاسها و رابطهای مخصوص پردازش اسناد XML را فراهم می کند.

فضا نام Microsoft

علاوه بر فضا نام System که در چارچوب .NET یافت می شود، میکروسافت چند فضا نام اضافه کرده است که برای زبان برنامه سازی ای که می خواهید از آن در برنامه کاربردی خود استفاده کنید عملکرد لازم را تامین می کنند. ممکن است شما بصورت مستقیم با این فضا نام کاری نداشته باشید.

- **Microsoft.VisualBasic**: این فضا نام محتوی CLR یا زمان اجرای Visual Basic.NET است. از این زمان اجرا با زبان Visual Basic.NET استفاده می شود. این فضا نام همچنین دربرگیرنده کلاس هایی است که از کامپایل و تولید کد با استفاده از زبان ویژوال بیسیک پشتیبانی می کنند.

• Microsoft.CSharp: این فضای نام دربرگیرنده کلاس هایی است که از کامپایل و تولید کد با استفاده از زبان C# پشتیبانی می کنند.

• Microsoft.JScript: این فضای نام دربرگیرنده کلاس هایی است که از کامپایل و تولید کد با استفاده از زبان JScript پشتیبانی می کنند.

• Microsoft.Win32: کلاسها و رابطهای مورد نیاز برای کار با کلیدها و hiveهای رجیستری را تامین می کند.

با وجود آنکه فضا نام ها از قبل تامین می شوند، می توانید برای استفاده از برنامه کاربردی ASP.NET فضا نام های خود را ایجاد کنید. برای هر کلاس ایجاد شده توسط سازنده یک فضا نام تولید می شود.

استفاده از فضا نام ها در صفحات ASP.NET

دو راه برای افزودن فضا نام به برنامه کاربردی ASP.NET وجود دارد. از شبه دستور (Directive) صفحه @Import برای صفحات ASPX و از کلمه کلیدی Imports برای افزودن فضا نام به برنامه codebehind مربوطه در ویژوال بیسیک استفاده می شود و برای زبان C# از دستور using استفاده می گردد. قطعه برنامه زیر ترکیب نحوی برای افزودن فضا نام System.Web.UI.WebControls به صفحه ASP.NET شما است.

```
<%@ Import namespace = "System.Web.UI.WebControls" %>
```

همین فضا نام را در قسمت codebehind بصورت زیر به برنامه اضافه می کنیم.

```
Imports System.Web.UI.WebControls (vb.net)
```

```
using System.Web.UI.WebControls; (C#)
```

(به تفاوت Import و Imports دقت کنید)

در صورتیکه می خواهید چند فضا نام را به صفحه ASP.NET خود و یا صفحه codebehind اضافه کنید باید هر کدام را جداگانه اضافه کنید. بعنوان مثال، برای افزودن فضا نام System.Web.UI.HtmlControls به صفحات با فضا نام های موجود، درست بعد از آخرین عبارت مهم به خط بعد بروید و Imports System.Web.UI.HtmlControls را اضافه کنید. به محض آنکه Imports System.Web.UI.HtmlControls را تایپ کنید، VS.NET فهرستی از فضا نام ها را ظاهر می کند، و می توانید به سادگی فضا نام مورد نظر را با ماوس برگزینید. امتیاز این فهرست آن است که مجبور نیستید همه فضا نام های NET را از حفظ بدانید، بلکه می توانید به آسانی از فهرست انتخاب کنید. این ویژگی با عنوان Intellisense شناخته می شود. برای صفحات ASP.NET از این ترکیب استفاده کنید:

```
<%@ Import namespace = "System.Web.UI.WebControls" %>
<%@ Import namespace = "System.Web.UI.HtmlControls" %>
<%@ Import namespace = "namespace name" %>
...
```


برای صفحات codebehind ویژوال بیسیک از این ترکیب استفاده کنید:

```
Imports System.Web.UI.WebControls  
Imports System.Web.UI.HtmlControls  
Imports namespace  
...
```

Smart Navigation چیست؟

مفهوم Smart Navigation و فواید آن

Smart Navigation یکی از بهترین ابزارهای جدیدی است که ASP.NET آنرا عرضه کرده است. این ابزار جدید باعث شده ظاهر برنامه های وب و احساسی که نسبت به آن وجود دارد شباهت بیشتری با برنامه های عادی و نوشته شده برای ویندوز پیدا کند.

یکی از موانع بزرگ برنامه های تحت وب به معماری و ساختار HTTP برمی گردد. جاییکه مجبوریم اطلاعات جمع آوری شده در سمت مشتری را به سرور بازگردانیم. به همین دلیل مجبور به رسم مجدد و کامل صفحه ای که قبلا دیده ایم می باشیم، که این نه تنها باعث می شود يك حالت فلش مانند در این رفت و برگشت و رسم مجدد رخ دهد، بلکه برای صفحه های بلند که برای دیدن تمام صفحه نیازمند به scrolling هستیم، باعث می شود که دیدمان را به اول صفحه انتقال دهد، چیزی که هم شاید دلخواه ما نباشد و هم اینکه ممکن است باعث سردرگمی کاربر گردد. همچنین این فرآیند باعث تغییر فوکوس کنترل ها و بسیاری از اتفاقات دیگر نیز می شود.

در برنامه های عادی ویندوز ما به طور معمول فقط قسمت هایی از صفحه را به روز می کنیم که تغییری در آن ایجاد شده باشد یا تحت تاثیر چیزی قرار گیرند و این بدون نیاز به تغییر در کل برنامه می باشد (مثلا فقط يك عضو به listbox ما اضافه می شود، بدون تغییر و رسم مجدد فرم برنامه).

Smart Navigation یا به عبارتی هدایت هوشمندانه این توانایی موجود در برنامه های ویندوز را برای برنامه های تحت وب فراهم می کند! اما قبل از هر چیز باید بدانید که این ابزار فقط برای IE می باشد و آن هم نسخه های ۵ به بالاتر آن. با این وجود شما می توانید Smart Navigation را فعال یا غیرفعال سازید، بدون آنکه تاثیری در برنامه شما بگذارد. حتی اگر شما در پروژه تان مرورگرهای مختلفی را مدنظر قرار داده باشید، می توانید Smart Navigation را فعال سازید. در این صورت ASP.NET نوع مرورگر را تشخیص داده و Smart Navigation را فقط برای مرورگرهای پشتیبانی شده فعال می سازد.

چهار مورد برجسته ای که Smart Navigation فراهم می کند عبارتند از:

- صفحه در میان درخواست ها يك نمایش ممتد را داراست و به عبارتی حالت فلش زدن به خود نمی گیرد.
- موقعیت Scroll را حفظ می کند.
- فوکوس عضو دارنده فوکوس را نگه می دارد.
- آخرین صفحه درون تاریخچه (History) نگهداری می شود.

این ابزار در حالت واقع گرایانه برای برنامه هایی که ارسال به عقب (!) Postback فراوانی دارند طراحی شده است ولی با توجه به این نکته که محتوای صفحه نباید زیاد تغییر نکند. احتمالا بنا به دلایل کارایی و نه اینکه در تغییرات زیاد ایرادی بهم بزند - مترجم. شاید يك چیز شگفت آور در مورد این ابزار این باشد که شما در حقیقت نیاز به نوشتن هیچ کد و برنامه ای ندارید.

نحوه استفاده

Smart Navigation درون هدایت کننده صفحه (Page directive : <% @%>)، برای تنظیم يك صفحه و درون web.config برای تنظیم کل برنامه استفاده می شود. برای تنظیم در Page Directive به صورت زیر عمل کنید:

<%@ Page SmartNavigation=true %>

و برای تنظیم در web.config از ساختار زیر استفاده نمایید:

```
<Configuration>  
<System.web>  
<Pages SmartNavigation=true />  
</System.web>  
</Configuration>
```

روش کار اینگونه است که کل صفحه بدرون يك فریم دورنی مخفی (hidden IFrame) بارگذاری (load) می شود و سپس فقط قسمت های تغییر کرده دوباره رندر (render) می شوند.

چرا به دات نت احتیاج داریم؟

به طور معمول نسل های جدید زبان های برنامه نویسی به این دلیل متولد می شوند که زبان های قدیمی تر دارای امکانات محدود بودند و یا قدرت استفاده از تکنولوژی های فعلی را به صورت مطلوب و ساده ندارند.

مهمترین نیازی که به عنوان آخرین تکنولوژی وجود دارد، برنامه نویسی در محیط اینترنت است. اینترنت در مدت تقریباً ۸ سال جای خود را به عنوان یکی از مهمترین وسایل ارتباطی برای کارهای روزمره و تجارت باز کرده است. سیستم های برنامه نویسی قدیمی تر امکان برنامه نویسی برای اینترنت را فراهم کرده بودند اما هر کدام دارای اشکالات بزرگی هستند، برای مثال تکنولوژی COM اولین بار در ویندوز به کار گرفته شد. در سال ۱۹۷۰ نیز سیستم هایی برای Unix نوشته شده بودند، جاوا نیز در اصل برای ابزارهای الکترونیکی بود و نه برای اینترنت.

سپس برای اولین بار یک سیستم جامع برای برنامه نویسی تحت اینترنت ایجاد شد. این سیستم -NET. از مراحل سطح پایین که به زبان ماشین می باشد تا بالاترین سطح که برنامه نویسی و ویژوال آن می باشد برای استفاده در اینترنت طراحی شده است. البته NET. فقط برای اینترنت نیست و با استفاده از آن می توان برنامه های کامل تحت Client نیز ایجاد کرد، اما بزرگترین مزیت آن دربرابر سیستم های دیگر امکانات اینترنت آن است.

برای اینکه مزایای استفاده از NET. را بهتر متوجه بشویم بهتر است در ابتدا معایب سیستم های پیشین را ذکر کنیم. شرکت مایکروسافت تا قبل از سال ۱۹۹۵ به برنامه نویسی در محیط های Client و Server می پرداخت، اما از آن سال به بعد توجه بیشتری به مساله برنامه نویسی در اینترنت کرد. مایکروسافت COM و COM+ را ایجاد کرد و آنها را در ویژوال استودیوی ۶ به کار گرفت. در سال ۱۹۹۹ حدود ۵۰ درصد از بزرگترین سایتهای تجارت الکترونیکی از محصولات مایکروسافت استفاده می کردند. اما هنوز هم مشکلات بزرگی در سیستم های مایکروسافت وجود داشت که یکی از آنها دشواری نوشتن برنامه در اینترنت با محصولات مایکروسافت بود. شرکت مایکروسافت برای راحتی کار برنامه نویس ها ASP یا Active Server Page را ایجاد کرد. با اینکه این یک قدم بزرگ بود و کارها را بسیار ساده کرد ولی هنوز از برنامه نویسی شی گرا پشتیبانی نمی کرد، همچنین در ویژوال استودیوی ۶ قسمتی برای Internet Application ایجاد شده بود و در آنها امکان ساختن Web Class وجود داشت ولی هیچ وقت به عنوان یک ابزار کار آمد برای برنامه نویسی وب در نظر گرفته نشد.

مدل برنامه نویسی DNA

مایکروسافت یک مدل برنامه نویسی به نام Distributed interNet Application دارد که بر پایه برنامه نویسی n-tier و COM بنا نهاده شده است. مدل DNA از سه بخش اساسی تشکیل شده است.

بخش اول به نام Presentation tire معروف است. در این بخش رابط تصویری کاربر وجود دارد و خود نیز به دو نوع Internet Browser و Win 32 GUI تقسیم می شود که هر کدام مشکلات خاص خود را دارند. در مدلی که از Win32 GUI یا همان نرم افزارهای معمولی استفاده می شود دو مشکل بزرگ وجود دارد؛ دشواری بروز رسانی نرم افزار و دیگری DLL Hell که در ادامه توضیح داده خواهد شد. در نوع دوم مشکلاتی از قبیل نبود امکانات برنامه نویسی کافی در محیط مرورگر، نبود رابط قوی با کاربر، نبودن مرورگرهای یکسان و... وجود دارد. همچنین همیشه یک اتصال به اینترنت یا اینترانت لازم است. در این نوع از برنامه نویسی می توان از Java Applet ها یا ActiveX استفاده کرد ولی مرورگر باید امکان استفاده از آن را داشته باشد، مخصوصاً هنگام استفاده از ActiveX که باید فقط از IE استفاده کرد.

بخش دوم که Middle tier نام دارد، مکانی است که اطلاعات و قوانین تجاری در آن وجود دارد. منظور از قوانین، متد ها و اجزائی هستند که اعمال کاربران را کنترل می کنند. مهمترین و آسان ترین زبان برای نوشتن این اجزا از DNA ویژوال بیسیک است. برنامه نویسی که بخواهد در این رده برنامه بنویسد باید آشنایی کاملی با COM و پروتکل های رایج داشته، همچنین باید مهارت کافی در استفاده از ADO و ADSI داشته باشد. مشخص است که یک اشتباه در این لایه باعث بروز خطا و نقص در کل سیستم می شود.

بخش سوم یا Data tier مکانی است که اطلاعات سازمان در آن ذخیره می شود. معمولاً در این قسمت از بانکهای پیشرفته رابطه ای مانند SQL Server و Oracle استفاده می کنند.

محدودیت های COM

همانطور که دیدید مهمترین قسمت در DNA همان COM است که در جای جای آن استفاده می شود. در اینجا برخی معایب COM ذکر می شود: (در ابتدای متن ذکر شد که برای درک نیاز به NET. باید ابتدا معایب سیستم های قدیمی را بشناسیم)

DLL Hell: اگر کوچکترین تغییری در یک COM ایجاد شود، دیگر برنامه هایی که از ورژن قبلی استفاده می کردند قادر به فعال ساختن نسخه جدید نیستند. هنگامی که در ویندوز، یک COM نصب شود برایش در رجیستری یک GUID ثبت می شود که اطلاعات آن COM را در خود ذخیره می کند. اگر یک برنامه از نسخه اول یک COM استفاده کند و بعد از مدتی شما تغییراتی در نسخه اول بدهید و بخواهید آن را دوباره در سیستم نصب کنید ویندوز به شما پیغام خطا می دهد چون ورژن آن تکراری است، اگر هم آن را به ورژن دوم ارتقا دهید نرم افزار قبلی هنوز به دنبال نسخه اول می گردد. این امر باعث می شود که شما مجبور شوید یکبار دیگر کل برنامه را کامپایل کرده و در کامپیوترتان نصب کنید.

کمبود در وراثت: در نسخه های COM که در حال حاضر هستند چیزی به نام وراثتی که در C++ وجود دارد نمی باشد، بلکه وراثت تنها در واسط یک COM می باشد، استفاده از آن هم چندان کمکی به برنامه نویسی نمی کند.

برخی محدودیت های برنامه نویسی اینترنتی در مدل DNA

۱- وجود دو محیط برنامه نویسی برای اینترنت و Client

نقصان در نوشتن برنامه هایی با رابط گرافیکی خوب که در اینترنت کار می کردند کاملاً مشهود است، نمونه بارز آن اختلاف در برنامه نویسی در ویژوال بیسیک و ASP است. ویژوال بیسیک با رابط گرافیکی کاملاً سطح بالا و ASP تقریباً رابط گرافیکی ندارد. همین امر باعث می شد که یک برنامه نویس مجبور باشد طیف وسیعی از تکنیک ها و زبان ها را فراگیرد تا بتواند برنامه ساده ای در اینترنت بنویسد.

۲- نبودن حالت های ذخیره اطلاعات رابط گرافیکی در صفحه های اینترنتی

نمونه این حالت زمانی است که در یک textbox متنی وجود داشته باشد. در برنامه های Win32 GUI متن داخل textbox تا زمانی که کاربر یا برنامه آن را تغییر نداده بر جای خود وجود دارد. اما در محیط اینترنت و نوع ASP با هر بار refresh کردن صفحه کل اطلاعات از بین می رود. البته این مشکل با استفاده از شی های Request و Response تقریباً قابل حل است ولی احتیاج به برنامه نویسی برای هر تکه از صفحه ASP دارد.

۳- نداشتن Event Handler در محیط برنامه نویسی اینترنت

یکی از مهمترین ابزارهای که در برنامه نویسی Win32 GUI وجود دارد استفاده از Event ها است. با تکنولوژی که در حال حاضر وجود دارد تنها راه رسیدن به این مهم استفاده از ActiveX است که

به علت مسایل امنیتی در بیش از ۹۵ درصد مواقع توسط کاربر استفاده از آن رد می شود.

معایب استفاده از API

API ها توابعی هستند که از ویندوز نسخه ۱ تا امروز در برنامه نویسی کاربرد داشته و دارند. مهمترین کاری که این توابع انجام می دهند انجام کارهای سخت و سطح پایین سیستمی است که احتیاج به برنامه نویسی زیادی دارند و یا حتی امکان ایجاد آن با زبان هایی مثل ویژوال بیسیک نیست. اما هر API از هر نسخه ویندوز تا نسخه دیگر آن می تواند دچار تغییرات بشود. برای مثال برنامه ای که در ویندوز ۹۸ نوشته شده باشد می تواند در ویندوز ۹۵ اجرا نشود. همچنین هم اکنون ابزارهای جدیدی به بازار آمده است که برای آنها نیز می توان برنامه نویسی کرد، مانند تلفن های سیار، کیوسک تلفن، دستگاه های کامپیوتری جیبی و غیره. در این نوع دستگاه ها دیگر ویندوز به مفهومی که در حال حاضر وجود دارد قابل اجرا نیست و در نتیجه API هم وجود ندارد. لازم به ذکر است که ویندوز CE برای دستگاه های مذکور می باشد ولی قابلیت های آن با ویندوزهای دیگر تفاوت زیادی دارد.

نشان دادن قابلیت‌های مرورگر در ASP.NET

اگرچه در حال حاضر جنگ مرورگرها تقریباً تمام شده است اما این موضوع دلیلی بر شناخته نشدن قابلیت‌های مرورگرها نیست. در اینجا توانایی ASP.NET در نشان دادن قابلیت‌های مرورگرها بحث شده است. بعنوان نمونه، مثال ۱ نوع مرورگر را به ما نشان می‌دهد.

```
<html><body>
You are using <% =Request.Browser.Type %>
</body></html>
```

برای نمونه اگر شما از IE 5 استفاده می‌کنید نتیجه خروجی چنین باید باشد:

You are using IE5

در مثال ۱ Request.Browser.Type یک رشته را که همان نام و نسخه‌ی مرورگر است را بر می‌گرداند. اما این موضوع چگونه صورت می‌گیرد؟

شیء HTTPBrowserCapabilities

در حقیقت خاصیت Browser در شیء Request کلاسی از HTTPBrowserCapabilities است که در فضا نام System.Web قرار دارد. وقتی که این کلاس روی یک صفحه ASP.NET نمونه سازی می‌شود خواص صفحه سرویس گیرنده ای را نشان می‌دهد که از آن برای اجرا شدن کد استفاده شده است. شیء Request در برگیرنده این خاصیت مرورگر است که این کلاس را میتوان معادل کلاس MSWC. BrowserCapabilities در ASP کلاسیک در نظر گرفت.

در لیست زیر اکثر خاصیت‌های شیء HTTPBrowserCapabilities تشریح شده است:

ActiveXControls: نشان می‌دهد که مرورگر اکتیو ایکس را ساپورت می‌کند یا نه.
AOL: چک می‌کند که مرورگر از نوع AOL است یا نه.
Cookies: نشان می‌دهد که مرورگر کوکی ها را ساپورت می‌کند یا نه باید توجه داشت که این خاصیت وضعیت فعال بودن یا غیر فعال بودن کوکی ها را نشان نمی‌دهد.
Crawler: نشان میدهد که مرورگر سرویس گیرنده از موتورهای جستجو تاثیر می‌پذیرد یا نه.
Browser: نوع مرورگر را نشان می‌دهد.
Frames: نشان می‌دهد که مرورگر از قابلیت Frame برخوردار است یا نه.
MajorVersion: نسخه اصلی مرورگر را نشان می‌دهد بعنوان مثال در IE5 عدد ۵ نشانگر نسخه اصلی است.
MinorVersion: نسخه جزئی (کوچکتر) مرورگر را نشان می‌دهد بعنوان مثال در IE5.1 عدد ۱. نشانگر نسخه جزئی است.
Type: نوع و نسخه مرورگر را بصورت یک رشته باز میگرداند..
VBScript: نشان می‌دهد که مرورگر VBScript را ساپورت می‌کند یا نه.
Version: نسخه اصلی و جزئی مرورگر را بعنوان یک رشته برمی‌گرداند.

در زیر نمونه کامل یک مثال آورده شده است.

```
<%@ page language="VB" %>
<%@ Import Namespace="System.Web" %>
<html>
<body>
```

```
<head><title>HTTPBrowserCapabilities Demo</title></head>
```

```
<%
```

```
Dim browserObj As HTTPBrowserCapabilities
```

```
browserObj = Request.Browser
```

```
%>
```

```
<font face="verdana, arial" size=2>
```

```
<p>Your browser supports ActiveX controls: <%=browserObj.ActiveXControls %>
```

```
</p>
```

```
<p>Your browser type: <%=browserObj.Type %> </p>
```

```
<p>Your browser version: <%=browserObj.Version%> </p>
```

```
... Add any other property that you would like to display
```

```
</font>
```

```
</body>
```

```
</html>
```


آشنائی با ASP.NET

NET نسل بعدی Active Server Pages یا ASP است که توسط شرکت میکروسافت ارائه شده است. این محصول توسط میکروسافت بعنوان شاخص اصلی فناوری در ساخت سایتهای وب در نظر گرفته شده است. با استفاده از ASP.NET می توان هم اینترنت کوچک یک شرکت را ساخت و هم یک سایت وب تجاری خیلی بزرگ را طراحی و پیاده سازی نمود. مهمترین نکاتی که در طراحی این محصول در نظر گرفته شده است راحتی استفاده و بالا بودن کارائی و قابلیت آن می باشد. در زیر برخی ویژگیهای ASP.NET را بررسی می کنیم.

- صفحات ASP.NET کامپایل می شوند. هنگامی که یک صفحه ASP.NET برای اولین بار توسط یک مراجعه کننده به سایت فراخوانی می شود، آن صفحه ابتدا کامپایل شده و بر روی سرور نگهداشته می شود و در فراخوانی های بعدی از آن استفاده می شود. این بدین معنی است که صفحات ASP.NET خیلی سریع اجرا می شوند.

- صفحات ASP.NET با ابزارهای روی سرور ساخته می شوند. با ابزارهای موجود در ASP.NET می توان صفحات پیچیده وب را براحتی طراحی نمود. بعنوان مثال با استفاده از ابزار DataGrid می توان به آسانی داده های موجود در یک بانک اطلاعاتی را تحت وب نمایش داد.

- مجموعه ASP.NET عضوی از بدنه NET است. بدنه NET دارای بیش از ۴۵۰۰ کلاس آماده جهت استفاده در ASP.NET است. این کلاس ها تقریباً هر نیازی را در برنامه نویسی برآورده می کنند. بعنوان مثال از این کلاس ها می توان جهت تولید تصاویر بر حسب تقاضا، به رمز درآوردن یک فایل و یا ارسال یک نامه استفاده کرد.

مقایسه ASP.NET و ASP کلاسیک

ASP.NET نسل بعدی ASP یا ASP کلاسیک است. اما این یک پیشرفت تکاملی است بطوریکه این دو فناوری تقریباً از یکدیگر متفاوتند. صفحات ASP با زبان های دستورالعمل نویسی مانند VBScript یا JScript ایجاد می شوند اما در ASP.NET ما یک فرایند کامل برنامه نویسی با زبانهای Visual Basic یا C# (سی-شارپ تلفظ شود) داریم. همچنین در ASP کلاسیک تنها پنج کلاس استاندارد (Request, Response, Application, Session, Server) وجود دارد حال آنکه در ASP.NET می توان از بیش از ۴۵۰۰ کلاس استاندارد موجود در بدنه NET، بهره جست. همچنین علیرغم قدرت و امکانات زیاد و متعدد ASP.NET، استفاده از آن در مقایسه با ASP کلاسیک بسیار آسانتر است. بعنوان مثال با استفاده از چند ابزار در یک صفحه ASP.NET می توان یک صفحه بسیار پیچیده HTML بدست آورد که ساخت آن در ASP کلاسیک ممکن است نیاز به چند روز کار داشته باشد.

زبانهای برنامه نویسی در ASP.NET

شما در ASP.NET می توانید از هر زبان برنامه نویسی که با بدنه NET سازگار باشد استفاده کنید. این زبانها عبارتند از Visual Basic.NET و C# و JScript.NET. این بدین معنی است که شما جهت نوشتن برنامه در ASP.NET نیاز به فراگیری زبان جدیدی ندارید و اگر یکی از زبانهای ویژوال بیسیک یا C++ یا جاوا را می دانید هم اکنون می توانید در ASP.NET برنامه بنویسید. از طرف دیگر تعدادی زبانهای دیگر توسط بعضی از شرکتهای فعال در این زمینه به مجموعه زبانهای استاندارد ASP.NET افزوده شده است. بعنوان مثال اگر مایل باشید حتی می توانید از PERL و COBOL هم در ASP.NET استفاده کنید.

ابزارهای ASP.NET

سالهاست که برنامه نویسان ویژوال بیسیک جهت ساخت فرم های خود از ابزارهای ویژوال بیسیک مانند TextBox و ListBox استفاده کرده اند. در ASP.NET هم شما می توانید از ابزارهای فراوان موجود در آن برای ساخت فرم ها و صفحات خود استفاده نمائید. در ASP.NET چهار دسته عمده از ابزارها موجود است:

- ابزارهای اصلی مانند TextBox، RadioButton، ListBox و Button.
- ابزارهای اعتباری برای حصول اطمینان از ورود و تأیید صحت اطلاعات ورودی فرم ها.
- ابزارهای داده ای برای ارتباط با بانک اطلاعاتی و دستکاری داده.
- ابزارهای پیشرفته جهت نمایش عناصر پیچیده در واسط کاربر مانند تقویم و آگهی های تبلیغاتی.

با استفاده از Visual Studio.NET شما براحتی می توانید با چیدن تصویری این ابزارها بر روی فرم مورد نظر، صفحه دلخواه خود را بسازید. در صورت تمایل حتی می توانید در یک ویرایشگر ساده متن مانند Notepad برنامه مورد نظر را نوشته و از این ابزارها استفاده کنید.

دریافت ASP.NET

جهت شروع برنامه نویسی در ASP.NET تنها کافی است که مجموعه ASP.NET را به همراه بدنه NET، از سایت میکروسافت دریافت کنید.

<http://msdn.microsoft.com/downloads/default.asp?URL=/code/sample.asp?url=/msdn-files/027/000/976/msdncompositedoc.xml>

ASP.NET با سیستم عامل های Windows 2000 (نسخه Server و Professional) و Windows XP کاملاً سازگار است.

نحوه پردازش صفحات ASP.NET بر روی سرویس دهنده وب

برنامه های وب از معماری سرویس گیرنده - سرویس دهنده تبعیت نموده و بر روی سرویس دهنده وب مستقر و مسئولیت پاسخگوئی به درخواست های ارسالی توسط سرویس گیرندگان را برعهده خواهند داشت. در سمت سرویس گیرنده ، مرورگر و در سمت سرویس دهنده ، سرویس دهنده وب دارای جایگاهی خاص می باشند . مرورگر ، میزبان برنامه وب بوده و مهمترین وظیفه آن ارائه بخش رابط کاربر یک برنامه وب است . در این راستا ، مرورگر دارای پتانسیل لازم به منظور تفسیر و نمایش تگ های HTML می باشد . در سمت سرویس دهنده ، برنامه های وب با نظارت و مدیریت یک سرویس دهنده وب (مثلاً " IIS) اجراء می گردند . سرویس دهنده وب ، مسئولیت مدیریت برنامه ، پردازش درخواست های ارسالی توسط سرویس گیرندگان و ارائه پاسخ لازم به سرویس گیرندگان را بر عهده دارد . به منظور قانونمند کردن ارسال درخواست سرویس گیرندگان و ارائه پاسخ سرویس دهنده ، می بایست از یک پروتکل ارتباطی خاص استفاده گردد. پروتکل ، مجموعه ای از قوانین لازم بمنظور تشریح نحوه ارتباط دو و یا چندین آیم از طریق یک محیط انتقال (زیر ساخت انتقال داده) نظیر اینترنت است . در برنامه های وب (ارسال درخواست توسط سرویس گیرنده و پاسخ به درخواست توسط سرویس دهنده) از پروتکل ارتباطی Hypertext Transport Protocol (HTTP) ، استفاده می گردد.

ASP.NET پلات فرم مایکروسافت برای طراحی و پیاده سازی برنامه های وب در دات نت می باشد . پس از درخواست یک صفحه ASP.NET توسط مرورگر سرویس گیرنده ، پردازش های متعددی بر روی سرویس دهنده وب به منظور ارائه پاسخ لازم ، انجام خواهد شد. شاید تاکنون سوالات مختلفی در رابطه با نحوه پردازش صفحات ASP.NET بر روی سرویس دهنده ، برای شما مطرح شده باشد :

پس از درخواست یک صفحه ASP.NET ، بر روی سرویس دهنده وب چه اتفاقی می افتد ؟
نحوه برخورد سرویس دهنده وب با درخواست ارسالی توسط سرویس گیرنده چگونه است ؟
تگ های HTML چگونه تولید و برای مرورگر ارسال می گردد؟
و شاید سوالات دیگر!

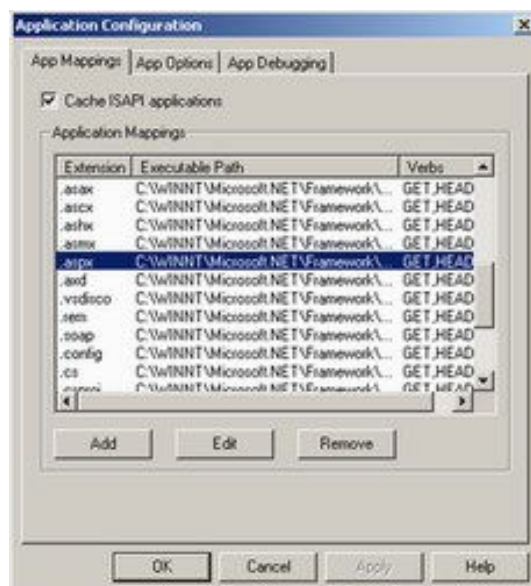
در این مقاله قصد داریم با نحوه پردازش صفحات ASP.NET بر روی سرویس دهنده بیشتر آشنا شویم . بدیهی است تشریح تمامی مراحل با ذکر جزئیات از حوصله یک مقاله خارج بوده و هدف آشنائی با کلیات موضوع با یک روند مشخص و سیستماتیک است .

مرحله اول : ایجاد یک درخواست HTTP برای یک صفحه ASP.NET توسط مرورگر
پردازش با درخواست یک صفحه ASP.NET که توسط مرورگر ایجاد می شود ، آغاز می گردد . مثلاً " یک کاربر ممکن است در بخش آدرس مرورگر کامپیوتر خود آدرس www.srco.ir//Articles/DocView.asp?ID=210 را به منظور دریافت این مقاله وارد نماید . مرورگر در ادامه یک درخواست HTTP را از سرویس دهنده وب محل استقرار سایت Srco.ir ایجاد و درخواست فایل حاوی مقاله را می نماید .

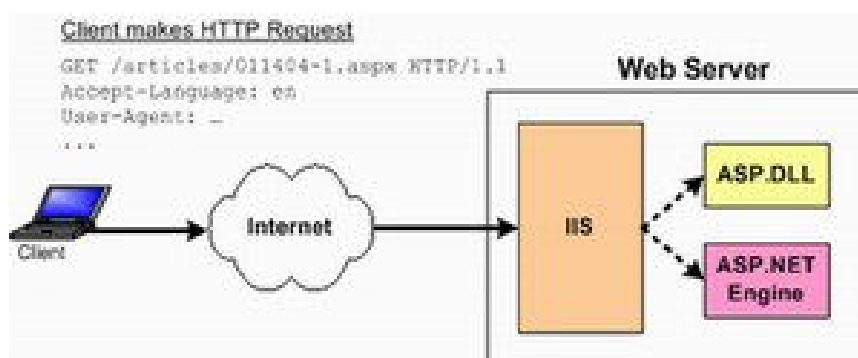
مرحله دوم : دریافت درخواست HTTP ، توسط سرویس دهنده وب
مهمترین وظیفه سرویس دهنده وب ، دریافت درخواست ارسالی HTTP و ارائه منبع درخواست شده در قالب یک پاسخ HTTP است . سرویس دهنده وب (مثلاً " IIS) ، پس از دریافت درخواست ارسال شده توسط سرویس گیرنده ، تصمیم لازم در رابطه با نحوه برخورد با آن را اتخاذ می نماید. محور تصمیم گیری فوق بر پایه نوع انشعاب فایل درخواستی استوار می باشد. مثلاً " در صورتیکه فایل درخواستی دارای انشعاب .asp ، باشد ، IIS درخواست را به سمت `asp.dll` هدایت تا عملیات مرتبط با آن انجام شود . انشعابات فایل متعددی به موتور ASP.NET ، مپ می گردند . برخی از آنان شامل موارد زیر می باشد :

انشعاب .aspx ، برای صفحات وب ASP.NET

انشعاب asmx ، ، براي سرويس هاي وب ASP.NET
 انشعاب config ، ، براي فايل هاي پيكربندي ASP.NET
 انشعاب ashx ، ، براي هندلرهاي سفارشي ASP.NET HTTP
 انشعاب rem ، ، براي منابع راه دور
 و ساير انشعابات ديگر



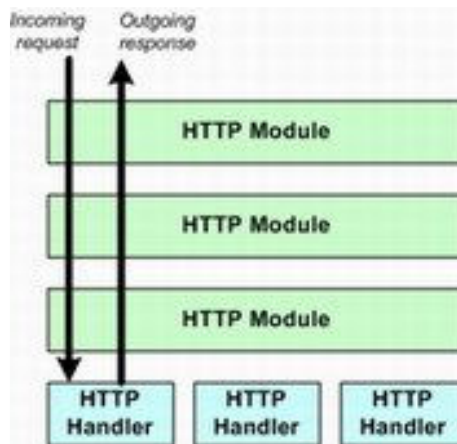
پس از دريافت درخواست ارسالي توسط سرويس گيرنده ، سرويس دهنده وب آن را در اختيار مسئول مربوطه قرار خواهد داد . مثلاً" در صورتيكه درخواست دريافتي مربوط به يك صفحه ASP كلاسيك باشد ، درخواست در اختيار asp.dll گذاشته شده و يا در صورتيكه درخواست در ارتباط با يك صفحه ASP.NET باشد ، درخواست در اختيار موتور ASP.NET قرار داده مي شود . همانگونه كه اشاره گرديد ، معيار اصلي در اين تصميم گيري ، نوع انشعاب فايل درخواست شده توسط سرويس گيرنده مي باشد . شكل زير مراحل اول و دوم اشاره شده را نشان مي دهد .



منبع : <http://www.4guysfromrolla.com>

مرحله سوم : عملکرد موتور ASP.NET
 پس از درخواست يك صفحه ASP.NET توسط سرويس گيرنده و دريافت آن توسط سرويس دهنده وب ، درخواست دريافتي در اختيار موتور ASP.NET قرار داده مي شود . از موتور ASP.NET ، اغلب با نام ASP.NET HTTP pipeline ياد مي گردد. علت نامگذاري فوق ، بدین دليل است كه درخواست دريافتي از بين تعداد متغيري از HTTP modules در بين مسير خود براي رسيدن به يك HTTP handler عبور مي نمايد . HTTP modules ، كلاس هائي مي باشند كه امكان دستيابي

به درخواست دریافتی را دارا می باشند. این ماژول ها قادر به بازبینی و بررسی درخواست دریافتی و اتخاذ تصمیماتی می باشند که مستقیماً بر نحوه گردش داخلی (روند برخورد با درخواست) تاثیر خواهد گذاشت . درخواست دریافتی پس از عبور از ماژول های مسخ شده HTTP ، به یک HTTP Handler خواهد رسید . HTTP Handler مسئولیت ایجاد خروجی لازم به منظور ارسال برای مرورگر متقاضی (ارسال کننده درخواست) را برعهده دارد. شکل زیر ، pipeline یک درخواست ASP.NET را نشان می دهد .



منبع : <http://www.4guysfromrolla.com>

تعداد زیادی از ماژول های HTTP از قبل ایجاد شده، بصورت پیش فرض در HTTP pipeline وجود دارد:

OutputCache ، مسئولیت برگرداندن و Caching خروجی صفحات HTML در صورت نیاز ، برعهده دارد .

Session ، ماژول فوق ، مسئولیت لود Session state را بر اساس درخواست دریافتی کاربر و روش Session که در فایل Web.config مشخص شده است ، برعهده دارد .

FormsAuthentication ، ماژول فوق ، مسئولیت تأیید کاربران بر اساس مدل تعریف شده Forms Authentication را در صورت ضرورت برعهده دارد .

و موارد دیگر

به منظور آشنایی با ماژول های پیش فرض، می توان مقادیر نسبت داده شده به عنصر `<httpModules>` در فایل `machine.config` را مشاهده نمود. جدول زیر مقدار پیش فرض عنصر `<httpModules>` را نشان می دهد .

machine.Config: httpModules Section

Path : \$WINDOWS\$\Microsoft.NET\Framework\\$VERSION\$\CONFIG

```
<httpModules>
<add name="OutputCache" type="System.Web.Caching.OutputCacheModule"/>
<add name="Session" type="System.Web.SessionState.SessionStateModule"/>
<add name="WindowsAuthentication"
type="System.Web.Security.WindowsAuthenticationModule"/>
```

```

<add name="FormsAuthentication"
type="System.Web.Security.FormsAuthenticationModule"/>
<add name="PassportAuthentication"
type="System.Web.Security.PassportAuthenticationModule"/>
<add name="UrlAuthorization" type="System.Web.Security.UrlAuthorizationModule"/>
<add name="FileAuthorization" type="System.Web.Security.FileAuthorizationModule"/>
<add name="ErrorHandlerModule" type="System.Web.Mobile.ErrorHandlerModule,
System.Web.Mobile,
Version=1.0.5000.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a"/>
</httpModules>

```

هاندلرهای HTTP ، نقطه پایان در ASP.NET HTTP pipeline می باشند . مسئولیت handler ، تولید خروجی برای منبع درخواست شده است . برای صفحات ASP.NET ، این به معنی Rendering ، کنترل های وب به HTML و برگرداندن HTML می باشد. برای یک سرویس وب ، مسئولیت فوق ، شامل اجرای متد مشخص شده و Wrapping مقایر برگردانده شده به یک پاسخ مناسب و با فرمت SOAP می باشد . منابع متفاوت ASP.NET از هاندلرهای متفاوت HTTP استفاده می نمایند. هاندلرهای پیش فرض استفاده شده ، توسط بخش <httpHandlers> فایل machine.config مشخص شده اند. بخش فوق ، شامل کلاس هایی است که یا خود HTTP handler بوده و یا HTTP handler factories ، می باشند. یک HTTP handler factory ، صرفاً یک نمونه از یک HTTP handler را پس از فراخوانی ، برمی گرداند . جدول زیر ، اطلاعات عنصر <httpHandlers> در فایل machine.config را نشان می دهد .

machine.Config: httpHandlers Section

Path : \$WINDOWS\$\Microsoft.NET\Framework\\$VERSION\$\CONFIG

```

<httpHandlers>
<add verb="*" path="trace.axd" type="System.Web.Handlers.TraceHandler"/>
<add verb="*" path="*.aspx" type="System.Web.UI.PageHandlerFactory"/>
<add verb="*" path="*.ashx" type="System.Web.UI.SimpleHandlerFactory"/>
. . .
<add verb="*" path="*.resources" type="System.Web.HttpForbiddenHandler"/>
<add verb="GET,HEAD" path="*" type="System.Web.StaticFileHandler"/>
<add verb="*" path="*" type="System.Web.HttpMethodNotAllowedHandler"/>
</httpHandlers>

```

لازم است به این نکته اشاره گردد که امکان ایجاد HTTP modules و HTTP handler اختصاصی ، توسط طراحان و پیاده کنندگان برنامه های وب ASP.NET نیز وجود دارد . پس از ایجاد مازول ها و هاندلرهای HTTP ، می توان آنان را به pipeline ملحق تا برای تمامی سایت های وب موجود بر سرویس دهنده وب ، قابل استفاده گردند. بدین منظور، می توان تغییرات لازم را در فایل machine.config اعمال تا زمینه استفاده از آنان توسط تمامی برنامه های وب فراهم گردد . در این رابطه می توان تغییرات را در فایل Web.config نیز اعمال نمود، در چنین مواردی امکان استفاده از مازول ها و هاندلرهای HTTP ایجاد شده ، صرفاً برای یک برنامه وب وجود خواهد داشت .

مرحله چهارم : تولید خروجی

آخرین مرحله در ارتباط با پردازش یک صفحه ASP.NET بر روی سرویس دهنده وب ، شامل ایجاد خروجی مناسب است . خروجی فوق ، در ادامه از طریق مازول های HTTP عبور داده شده تا

مجدداً به IIS برسد . در نهایت IIS ، خروجی تولید شده را برای سرویس گیرنده متقاصی ارسال می نماید . مراحل لازم به منظور تولید خروجی با توجه به HTTP handler متفاوت بوده و در ادامه صرفاً یک حالت خاص آن را بررسی می نمائیم (هندلر HTTP که از آن به منظور rendering صفحات ASP.NET استفاده می گردد).

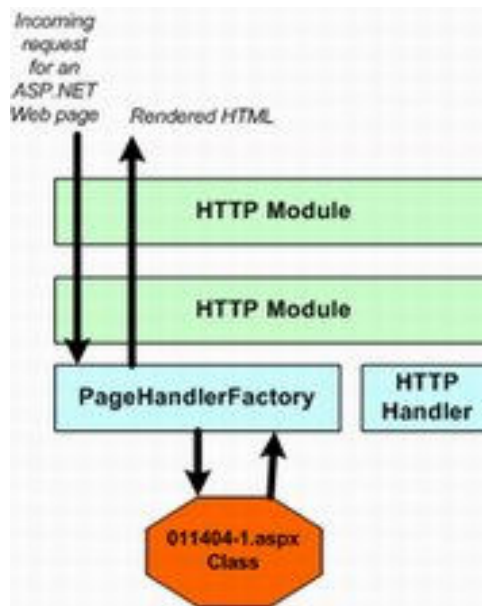
سرویس دهنده وب (IIS) پس از دریافت درخواستی برای یک صفحه ASP.NET (انشعاب فایل .aspx) ، آن را در اختیار موتور ASP.NET ، قرار خواهد داد. درخواست دریافتی در ادامه از بین مازول ها عبور داده شده تا به PageHandlerFactory برسد (در بخش <httpHandlers> فایل machin.config که قبلاً mapping آن انجام شده است) .

machine.Config: httpHandlers Section :PageHandlerFactory
Path : \$WINDOWS\$\Microsoft.NET\Framework\\$VERSION\$\CONFIG

```
<httpHandlers>
...
<add verb="*" path="*.aspx" type="System.Web.UI.PageHandlerFactory"/>
. ...
</httpHandlers>
```

کلاس PageHandlerFactory ، یک HTTP handler factory است که وظیفه آن ارائه نمونه ای از یک HTTP handler بوده که قادر به برخورد مناسب با درخواست ارسالی است. مهمترین رسالت PageHandlerFactory ، یافتن کلاس ترجمه شده ای است که نشاندهنده صفحه ASP.NET درخواستی می باشد. در صورتیکه از ویژوال استودیو دات نت به منظور ایجاد صفحات ASP.NET استفاده می گردد ، صفحات وب از دو فایل جداگانه (یک فایل با انشعاب .aspx ، شامل صرفاً کنترل های وب و تگ های HTML و یک فایل .aspx.vb یا .aspx.cs شامل کلاس code-behind) کد سمت سرویس دهنده () ، تشکیل می گردند. در صورتیکه از ویژوال استودیو دات نت استفاده نمی گردد ، می توان از یک بلاک سمت سرویس دهنده <Script> استفاده تا کد سمت سرویس دهنده را در خود نگهداری نماید . صرفنظر از اینکه از کدام رویکرد استفاده می گردد ، زمانیکه صفحه ASP.NET اولین مرتبه و پس از ایجاد تغییر در تگ های HTML و یا محتوی کنترل وب ، مشاهده می گردد ، موتور ASP.NET یک کلاس که مشتق شده از کلاس System.Web.UI.Page می باشد را ایجاد می نماید . کلاس فوق بصورت اتوماتیک ایجاد و کمپایل می گردد .

Page Class ، عملیات پیاده سازی IHttpHandler را انجام خواهد داد. PageHandlerFactory ، در ادامه بررسی لازم در خصوص وجود یک نسخه کمپایل شده از صفحه ASP.NET درخواستی را انجام خواهد داد. در صورتیکه صفحه ترجمه شده وجود نداشته باشد ، PageHandlerFactory آن را بصورت پویا ایجاد و ترجمه خواهد کرد . کلاس فوق ، در ادامه متد خاصی را به منظور تولید HTML ، فرا می خواند . اطلاعات تولید شده به فرمت HTML ، در نهایت برای سرویس گیرنده ارسال می گردد. وجود تاخیر در مشاهده صفحات ASP.NET که بر روی آنان تغییراتی اعمال شده است (HTML و یا محتوی کنترل وب) ، بدین دلیل است که موتور ASP.NET نیازمند ایجاد و ترجمه مجدد کلاس مرتبط با صفحه ASP.NET می باشد.



منبع : <http://www.4guysfromrolla.com>

پس از ایجاد و ترجمه کلاس توسط PageHandlerFactory ، امکان فراخوانی کلاس ایجاد شده به منظور تولید HTML ، فراهم می گردد . فرآیند Rendering که شامل بدست آوردن HTML لازم برای صفحه ASP.NET درخواست شده می باشد از حوصله این مقاله خارج بوده و می توان در این رابطه از مقاله The ASP.NET Page Object Model استفاده نمود .

ارسال ایمیل در ASP.NET با استفاده از HTML Template

آیا تاکنون سعی کرده اید برای سایت خود خبرنامه ایجاد کنید؟ آیا تاکنون وسوسه شده اید که سیستمی طراحی کنید که در صورتی که بینندگان سایت شما نظرات خود را در سایت شما وارد کنند برای آنها یک Email تشکرآمیز ارسال کنید؟ آیا می دانید ساختن یک HTML Template زیبا برای خبرنامه شما و استفاده همیشگی از آن جهت ارسال خبرنامه در ASP.NET بسیار ساده صورت می گیرد؟ چنانچه سوالات بالا شما را به دانستن بیشتر ترغیب کرده است به شما تبریک می گویم! چرا که در ادامه این مقاله شما روش بسیار ساده ارسال Email های HTML زیبا را بدون اینکه نگران چگونگی کدنویسی HTML متغیر رشته ای Body در MailMessage باشید را فرا خواهید گرفت.

برای شروع بیا بید به روش کار نظری بیانداریم: خوب احتمالا شما با روش ساختن یک نمونه از آبجکت MailMessage که در ASP.NET برای ارسال Email استفاده می شود آشنایی دارید (اگر این چنین نیست در ادامه مقاله به طور اجمالی توضیحاتی ارائه شده است) تنها قسمتی که باید مورد توجه قرار دهید این است که چگونه می توانیم Body نامه خود را به فرمت HTML و بدون نیاز به اینکه تمامی تگها را پشت سر هم در یک رشته طولانی و سردرگم کننده تایپ کنیم، درآوریم. یک ایده جالب این است که ما Template نامه خود را با استفاده از ادیتورهای WYSIWYG مثل FrontPage یا Dreamweaver طراحی کنیم و سپس تمامی کدهای HTML آن را درون یک متغیر رشته ای Import کرده و از آن استفاده کنیم. این کار بسیار آسان است. اما می توان این نامه را برای هر کاربر کمی سفارشی (Customize) کرد! برای مثال شما می توانید در ابتدای نامه کاربر را با نام وی مورد خطاب قرار دهید. بهتر است از این پس توضیحات را همراه با کدنویسی دنبال کنیم. (کلیه کدهای Server Side به زبان VB.NET نوشته شده است) سه گام اساسی برای این کار وجود دارد:

۱- ساختن Template مورد نظر شما جهت ارسال Email:

برای این کار کافی است که Template مورد نظر خود را به فرمت HTML طراحی کنید. این به خود شما بستگی دارد که کدهای HTML را به صورت دستی بنویسید و یا از ادیتورهای WYSIWYG مانند FrontPage یا Dreamweaver یا GoLive! استفاده کنید. تنها تفاوتی که در اینجا وجود دارد این است که شما بایستی قسمتهایی از متن نامه خود را که متغیر هستند (مانند نام گیرنده Email، آدرس پست الکترونیکی وی و ...) را به گونه ای از بقیه قسمتها متمایز کنید. برای مثال اگر شما می خواهید در ابتدای نامه، دریافت کننده نامه را با نام خود مورد خطاب قرار دهید عبارت را به صورت زیر وارد کنید: "سلام" #NAME# لطفا توجه کنید که هیچ محدودیتی در تکنیک به کار رفته وجود ندارد و قرار دادن کاراکترهای ## فقط جهت متمایز ساختن این قسمت از محتوای استاتیک صفحه است و شما می توانید به صورت دیگر آن را مشخص کنید مثلا NAME? یا هر چیز مشابه دیگر. این قسمت بعدا با نام شخص مورد نظر ما جایگزین می شود. نکته قابل ذکر دیگر اینکه چنانچه قصد دارید نامه خود را فارسی ارسال کنید تگ زیر را فراموش نکنید:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

۲- خواندن فایل Template جهت قراردادن در Body نامه شما:

پس از اینکه Template را آماده نمودید بایستی این فایل را بخوانید، قسمتهای مورد نظر (متمایز شده با ##) را با عبارات مورد نظر خود (که می توانید از Database استخراج کنید) جایگزین کنید و این متن را به صورت یک متغیر رشته ای در Body نامه خود قرار دهید (گام ۳). این کار به وسیله کد زیر صورت می پذیرد: (قبل از هر چیز شما بایستی Namespace مورد نیاز که در اینجا System.IO می باشد را Import کنید)

Imports System.IO

Dim reader As StreamReader

Dim strFileName As String = Server.MapPath("templatel.htm")

Dim strFileText

reader = File.OpenText(strFileName)

While reader.Peek <> -1

strFileText += reader.ReadLine()

End While

reader.Close()

strFileText = Replace(strFileText, "#Username#", Name.Text)

strFileText = Replace(strFileText, "#MsgID#", Request("MsgID"))

strFileText = Replace(strFileText, "#AuthorID#", Request("AuthorID"))

حال بیا باید نگاهی به کد فوق بیاندازیم: در سطر اول شما یک شی از نوع StreamReader که جهت خواندن فایل‌های متنی به کار می‌رود را می‌سازید و در سطر بعدی مسیر فیزیکی فایل Template خود را در متغیر رشته‌ای strFileName ذخیره می‌کنید.

نکته: در صورتیکه بخواهید در فایل کلاس خود از کد فوق استفاده کنید بایستی مسیر فیزیکی فایل را با استفاده از "System.Web.HttpContext.Current.Server.MapPath("template.htm") به دست آورید. اما در Code Behind یک WebForm همان Server.MapPath() کافی است.

سپس در سطر بعدی ما با استفاده از متد OpenText فایل مورد نظر را باز کرده و متن آن را در Reader قرار می‌دهیم. حال در یک حلقه While...End While سطر به سطر فایل Template را (که در اینجا کدهای HTML ما هستند) می‌خوانیم و در متغیر strFileText ذخیره می‌کنیم. Reader.Peek <> -1 بررسی می‌کند که آیا به انتهای فایل رسیده ایم یا خیر. پس از اینکه کل فایل را خوانده و در متغیر ذخیره کردیم، شی reader را می‌بندیم.

در اینجا شما باید Template نامه خود را برای کاربر خاص Customize کنید. یعنی عبارات محصور شده با ## را با اطلاعات کاربر (که از بانک اطلاعاتی استخراج می‌شود و یا از یک Query String دریافت می‌شود) جایگزین کنید. در مثال بالا مقدار اول در Template با مقدار یک Textbox و دو مورد بعدی با Query String های انتهای یک URL جایگزین می‌شود.

حاصل کار یک متغیر رشته ای به نام strFileText است که حاوی کد HTML مورد نیاز شما برای ارسال یک HTML Email می‌باشد. که مثلا عبارت ابتدای آن به صورت زیر در آمده است: "سلام مهدی"

۳- استفاده از قالب Import شده فوق و ارسال Email:

حال به آسانی Email را ساخته و ارسال می‌کنیم. بدین ترتیب: (Imports System.Web.Mail) فراموش نکنید!

'Creating and sending mail to user

Dim objMail As New MailMessage()

objMail.From = "You@YourDomain.com"

```
objMail.To = Email.Text.Trim  
objMail.Subject = "YOUR SUBJECT GOES HERE..."  
objMail.BodyFormat = MailFormat.Html  
objMail.BodyEncoding = System.Text.Encoding.UTF8  
objMail.Body = strFileText  
SmtpMail.SmtpServer = "smtp.YOURSERVER.com"  
SmtpMail.Send(objMail)
```

این قسمت نیاز به توضیح چندانی ندارد. جز اینکه objMail.To را بایستی به صورت داینامیک (از DB و ...) تغییر دهید (در اینجا مقدار از یک Textbox دریافت شده است)، SMTP Server خود را مشخص کنید و Email را ارسال کنید! به همین سادگی. بقیه موارد بسیار واضح هستند.

لطفا به این نکته توجه کنید که برخی SMTP Server هایی که نیاز به Authentication دارند را نمی توان برای این منظور (ارسال Email به آدرس های خارج از SMTP Server فوق) مورد استفاده قرار داد. در این خصوص لطفا به Administrator سرور خود رجوع کنید.

اضافه کردن تصویر به پایگاه داده

در بسیاری مواقع ما نیاز به افزودن تصویر به پایگاه داده داریم. از آنجایی که دسترسی به فایل‌های موجود در سرور جهت استفاده هکرها بسیار سهل الوصول است، برخی مواقع نمی‌توانیم اطلاعات حساس را در قالب فایل‌های تصویری در سرور ذخیره کنیم. به همین جهت بهتر آن است که در قالب نوع داده image در بانک اطلاعاتی ذخیره شود.

در این قسمت موارد زیر بررسی می‌شوند:

بررسی پیش نیازهای اضافه نمودن فایل تصویری
کار با شیء Stream
پیدا کردن اندازه و نوع تصویری که آپلود می‌شود
چگونگی استفاده از متد InputStream

بررسی پیش نیازهای اضافه نمودن فایل تصویری

دو مورد اصلی که قبل از شروع آپلود نیاز داریم عبارتند از:

۱. - تنظیم خاصیت enctype فرم مربوطه (در برجسب Form) به enctype="multipart/form-data"

- داشتن <input type=file> که به کاربر اجازه انتخاب فایل مورد نظرش را می‌دهد. (فایلی که در پایگاه داده باید ذخیره شود) و یک Submit button جهت اجرای عملیات مربوط به آپلود کردن فای

- استفاده از فضا نام System.IO جهت سروکار داشتن با شیء Stream

۲. سه پارامتر بالا در یک صفحه aspx بکار می‌روند. ما همچنین به پیش نیازهای زیر در SQL Server نیازمندیم:

- داشتن حداقل یک جدول با فیلدی از نوع image

- بهتر است که فیلد دیگری از نوع varchar جهت ذخیره تایپ تصویر داشته باشیم

روند کار بدین صورت است که در ابتدا محتویات فایل تصویر خوانده می‌شود و سپس تصویر به جدول افزوده می‌شود. در زیر کد مربوط به رویداد OnClick مربوط به Submit button که تصویر را خوانده و به جدول SQL اضافه می‌کند، می‌پردازیم:

```
Public Sub AddPerson(sender As Object, e As EventArgs)
```

```
Dim intImageSize As Int64
```

```
Dim strImageType As String
```

```
Dim ImageStream As Stream
```

```
' Gets the Size of the Image
```

```
intImageSize = PersonImage.PostedFile.ContentLength
```

```
' Gets the Image Type
```

```
strImageType = PersonImage.PostedFile.ContentType
```

```
' Reads the Image
```

```
ImageStream = PersonImage.PostedFile.InputStream
```

```

Dim ImageContent(intImageSize) As Byte
Dim intStatus As Integer
intStatus = ImageStream.Read(ImageContent, 0, intImageSize)

' Create Instance of Connection and Command Object
Dim myConnection As New SqlConnection(
ConfigurationSettings.AppSettings("ConnectionString"))
Dim myCommand As New SqlCommand("sp_person_isp", myConnection)

' Mark the Command as a SPROC
myCommand.CommandType = CommandType.StoredProcedure

' Add Parameters to SPROC
Dim prmPersonImage As New SqlParameter("@PersonImage", SqlDbType.Image)
prmPersonImage.Value = ImageContent
myCommand.Parameters.Add(prmPersonImage)

Dim prmPersonImageType As New SqlParameter("@PersonImageType",
SqlDbType.VarChar, 255)
prmPersonImageType.Value = strImageType
myCommand.Parameters.Add(prmPersonImageType)

Try
    myConnection.Open()
    myCommand.ExecuteNonQuery()
    myConnection.Close()
    Response.Write("New person successfully added!")
Catch SQLexc As SqlException
    Response.Write("Insert Failed. Error Details are: " & SQLexc.ToString())
End Try
End Sub

```

شیء PersonImage نام کنترل HTMLInputFile است. در ابتدا ما نیاز داریم که اندازه تصویری که جهت اضافه کردن انتخاب شده را بدانیم که توسط کد زیر قابل اندازه گیری است:

```
intImageSize = PersonImage.PostedFile.ContentLength
```

سپس نوع تصویر را با استفاده از خاصیت ContentType دریافت می کنیم. حال باید یک استریم از تصویر مربوطه دریافت شود:

```
ImageStream = PersonImage.PostedFile.InputStream
```

در اینجا ما یک آرایه از بایتها بنام ImageContent که آماده نگهداری محتویات تصویر است، داریم. تصویر ورودی توسط متد Read از شیء Stream خوانده می شود. متد Read سه آرگومان می گیرد:

۱. موقعیت مکانی که محتویات تصویر در آن باید کپی شود
۲. مکان شروع جهت خواندن
۳. تعداد بایتهایی که باید خوانده شود

و دستور مربوط به خواندن هم به شکل زیر است:

```
intStatus = ImageStream.Read(ImageContent, 0, intImageSize)
```

حالا ما می خواهیم محتویات تصویر ورودی را که خوانده ایم به یک جدول SQL اضافه کنیم. به همین منظور از یک زیرروال ذخیره شده که تصویر و نوع آن را به یک جدول اضافه می کند، بهره می بریم. حالا ما می خواهیم محتویات تصویر ورودی را که خوانده ایم به یک جدول SQL اضافه کنیم. به همین منظور از یک زیرروال ذخیره شده که تصویر و نوع آن را به یک جدول اضافه می کند، بهره می بریم.

فایل aspx مربوطه:

```
<%@ Page Language="vb" %>
<%@ Import Namespace="System.Data" %>
<%@ Import Namespace="System.Data.SqlClient" %>
<%@ Import Namespace="System.IO" %>
<html>
  <head>
    <title>Inserting Image to a SqlServer</title>
    <script runat=server>
      Public Sub AddPerson(sender As Object, e As EventArgs)
        ' above code
      End Sub
    </script>
  </head>

  <body>

    <form enctype="multipart/form-data" runat="server">
      <input type="file" id="PersonImage" runat=server />
      <asp:Button Text="Add Person" OnClick="AddPerson" Runat=server />
    </form>

  </body>
</html>
```

جدول و زیرروال استفاده شده:

Drop Table Person
Go

Create Table Person
(
PersonID Int Identity,
PersonEmail Varchar(255),
PersonName Varchar(255),

```

PersonSex Char(1),
PersonDOB DateTime,
PersonImage Image,
PersonImageType Varchar(255)
)

```

```

Drop Proc sp_person_isp
Go

```

```

Create Proc sp_person_isp
@PersonEmail Varchar(255),
@PersonName Varchar(255),
@PersonSex Char(1),
@PersonDOB DateTime,
@PersonImage Image,
@PersonImageType Varchar(255)
As
Begin
    Insert into Person
        (PersonEmail, PersonName, PersonSex,
        PersonDOB, PersonImage, PersonImageType)
    Values
        (@PersonEmail, @PersonName, @PersonSex,
        @PersonDOB, @PersonImage, @PersonImageType)
End

Go

```

بازیابی تصاویر از SQL Server در ASP.NET:
در مقایسه با اضافه کردن تصویر، بازیابی آن بسیار ساده است. تنها چیز جدید برای این قسمت استفاده از متد BinaryWrite موجود در شیء Response است.

کدهای مورد استفاده جهت بازیابی تصاویر از SQL Server:

```

Public Sub Page_Load(sender As Object, e As EventArgs)

    Dim myConnection As New SqlConnection(
    ConfigurationSettings.AppSettings("ConnectionString"))
    Dim myCommand As New SqlCommand("Select * from Person", myConnection)

    Try
        myConnection.Open()
        Dim myDataReader as SqlDataReader
        myDataReader =
myCommand.ExecuteReader(CommandBehavior.CloseConnection)

        Do While (myDataReader.Read())

```

```

        Response.ContentType = myDataReader.Item("PersonImageType")
        Response.BinaryWrite(myDataReader.Item("PersonImage"))
    Loop

    myConnection.Close()
    Response.Write("Person info successfully retrieved!")
Catch SQLExc As SqlException
    Response.Write("Read Failed : " & SQLExc.ToString())
End Try
End Sub

```

تمام کار این مرحله اجرای یک دستور SQL و تکرار آن برای همه ردیف‌هاست. قبل از نمایش تصویر، نوع آن هم مشخص می شود.

بررسی کوکی‌ها (Cookies) و جلسات (Sessions) در ASP.NET

در این مقاله ابتدا به بررسی کوکی‌ها (Cookies) پرداخته و سپس جلسات (Sessions) را بررسی خواهیم کرد. به دلیل نزدیکی بسیار زیاد این دو مفهوم تصمیم گرفتیم همه آنها را در یک مقاله جمع‌آوری کنم. اگر مفاهیم فوق را به درستی درک کنید و طرز استفاده از آنها را یاد بگیرید، می‌توانید به قدرت برنامه‌های کاربردی وب خود بیفزایید. اما استفاده نابجا از این مفاهیم در وب سایت‌های مختلف می‌تواند به شدت روی عملکرد سایت تاثیر گذاشته و از مخاطبان و کارایی آن بکاهد. در نتیجه با مطالعه این مقاله در مورد این دو مفهوم، باید بتوانید تصمیم بگیرید که کدامیک برای وب سایت شما مناسب‌تر می‌باشد. در هر حال مثال‌هایی را در این مقاله خواهید دید که به طور حتم به شما کمک زیادی خواهند کرد. لازم به ذکر است که بیشتر سعی در انتقال مفاهیمی کرده‌ام که پایه کار نمی‌باشند و سعی کردم به بیان نکاتی پردازم که راحت پیدا نمی‌شوند.

کوکی‌ها (Cookies)

استفان والتر در کتاب ASP.NET Unleashed در ابتدای بخش کوکی‌ها اینگونه می‌گوید: "پروتکل HTTP هیچ امکانی را در اختیار وب سرور قرار نمی‌دهد تا بتواند به کمک آن تشخیص دهد درخواست جدید از همان مرورگری صادر شده که درخواست قبلی را فرستاده یا از مرورگر دیگری آمده است. از این جهت به HTTP صفت ناپایداری (Stateless) را می‌دهند. از نقطه نظر وب سرور هر درخواستی که برای دریافت یک صفحه صادر شده است از طرف کاربری جدید ارسال شده است." این به طور قطع آن چیزی نیست که ما می‌خواهیم! وقتی می‌خواهیم اطلاعات کاربر را در هر صفحه به او نشان بدهیم (از قبیل شناسه کاربری، کلمه عبور، سید خرید و...) باید بتوانیم وضعیت آن را حفظ کنیم یکی از راه‌های بسیار خوب در این زمینه استفاده از کوکی‌ها می‌باشد.

اولین بار Netscape کوکی‌ها را در مرورگر خود به کار برد و به تدریج کنسرسیوم وب (W3C) نیز آن را پذیرفت و امروزه اکثر مرورگرها از کوکی‌ها پشتیبانی می‌کنند. بر اساس مستندات اولیه Netscape، یک کوکی نمیتواند حجمی بیشتر از ۴ کیلوبایت داشته باشد و با بستن صفحه مرورگر کوکی‌ها نیز از بین می‌روند. البته نگران نباشید اینها کوکی‌هایی هستند که پارامتر Expires آنها تنظیم نشده است. اما اگر این پارامتر را تنظیم کنید، کوکی‌ها باقی مانده و دائمی می‌شوند. اما تا کی؟ تا آن تاریخی که در خاصیت Expires تنظیم کرده‌اید. مرورگرهایی که می‌توانند با کوکی‌ها کار کنند دارای چند فایل ویژه می‌باشند که در ویندوز به آنها فایل‌های کوکی و در مکینتاش فایل‌های جادویی می‌گویند. کوکی‌ها از طریق هدرهای HTTP بین مرورگر و سرور جابجا می‌شوند. سرور با استفاده از هدر Set Cookie یک کوکی جدید ایجاد کرده و در درخواست‌های بعدی این کوکی به سرور فرستاده می‌شود.

در مقاله‌ای از سایت ASPFREE در مورد خواندن و نوشتن کوکی‌ها اینگونه نوشته شده است: "برای نوشتن کوکی یک شیء جدید HttpCookie بسازید و مقدار یک رشته را به آن اختصاص دهید (به خاصیت Value آن) و سپس متد Add() را در Response.Cookies فرا بخوانید. شما همچنین می‌توانید مقدار Expires را به یک مقدار تاریخ تغییر دهید تا زمان انقضاء برای کوکی‌تان تعیین کرده باشید."

باید توجه داشته باشید که کوکی‌ها فقط مقادیر رشته‌ای را ذخیره می‌کنند و برای نوشتن مقادیر دیگر در کوکی‌ها باید هر آنها را به یک رشته تبدیل کنید. این کد از سایت CodeToad برای یادگیری نحوه استفاده کوکی‌ها بسیار مناسب می‌باشد:

```
Using System.Web;
```

// نوشتن

```
Response.Cookies["BackgroundColor"].Value = "Red";
```

// خواندن

```
Response.Write(Request.Cookies["BackgroundColor"].Value);
```

به دلایل امنیتی شما می‌توانید فقط کوکی‌هایی را بخوانید که از یک دامنه آمده باشند. همچنین ممکن است شما نیاز به کوکی‌هایی داشته باشید که چند آیتم را در خود نگهداری کنند، یک مثال برای این کار در زیر می‌بینید:

```
HttpCookieCollection cookies = Request.Cookies;
```

```
for (int n = 0; n < cookies.Count; n++) {  
    HttpCookie cookie = cookies[n];  
    Response.Write("<hr/>Name: <b>" + cookie.Name + "</b><br />");  
    Response.Write("Expiry: " + cookie.Expires + "<br />");  
    Response.Write("Address1: " + cookie.Address1 + "<br />");  
    Response.Write("Address2: " + cookie.Address2 + "<br />");  
    Response.Write("City: " + cookie.City + "<br />");  
    Response.Write("Zip: " + cookie.Zip + "<br />");  
}
```

یک مثال درباره کوکی‌های تو در تو به زبان VB.NET:

```
If Request.Form("savecookie") = "Yes" and ValidLogin = "Yes" Then  
    Response.Cookies("member")("username") = Request.Form("username")  
    Response.Cookies("member")("password") = Request.Form("password")  
    Response.Cookies("member").Expires = DATE + 365  
End if
```

جدول زیر بعضی از خصوصیات پیشرفته کوکی‌ها را نمایش می‌دهد:

خاصیت	توضیحات
Domain	دامنه‌ای که محدوده کوکی را تعیین می‌کند.
Path	مسیر منتسب به کوکی.
Secure	مقدار بولینی که تعیین می‌کند آیا کوکی باید فقط روی یک اتصال رمز شده ارسال گردد یا نه؟
HasKeys	مقدار بولینی که تعیین می‌کند که آیا کوکی مربوط به یک کوکی دیکشنری است یا نه؟

بطور خلاصه، کوکی‌ها چه خوبی‌هایی دارند؟ اشغال فضا روی کلاینت که باعث کاهش ترافیک و اشغال فضا روی سرور می‌شود و اطمینان کار. کوکی‌ها چه بدیهایی دارند؟ بعضی از مرورگرها از کوکی‌ها پشتیبانی نمی‌کند، بعضی از کاربران کوکی‌ها را پاک می‌کنند یا نمی‌پذیرند و این که فقط داده نوع رشته‌ای را ذخیره می‌کنند.

جلسات (Sessions)

فریم ورک دات نت برای رد گیری حرکت کاربر ما را تنها نگذاشته و یک امکان خوب به نام Session State را در اختیار ما قرار داده است. به طور پیش فرض وقتی کاربر اولین بار صفحه‌ای را از یک وب سایت ساخته شده با ASP.NET درخواست می‌کند یک کوکی جلسه به نام ASP.NET_SessionID ساخته شده و به مرورگر او ارسال میشود. با این کار ASP.NET قادر به پیگیری کاربر شده و میتواند در درخواست‌های بعدی او را شناسایی کند.

بر این اساس در ASP.NET یک شیء به نام Session قرار داده شده است که میتوانید از آن برای نگهداری اطلاعات مربوط به هر کاربر استفاده کنید. برای مثال دستور زیر یک آیتم با نام MyItem ایجاد کرده و Hello را به آن نسبت میدهد:

```
Session("MyItem")="Hello!"
```

هنگام کار با Sessionها باید به نکات زیر توجه کنید:

۱. هر Session اگر کاربر مرورگر را ببندد یا ۲۰ دقیقه از سرور درخواست نکند از بین می‌رود.

۲. Session هر کاربر جدا از Session بقیه کاربران است.

۳. در Session بر خلاف کوکی‌ها می‌توان شیء هم ذخیره کرد.

جدول زیر بعضی از خصوصیات و متدهای شیء Session را نمایش میدهد:

توضیحات	خاصیت/متد
پاک کردن Session	Remove
پاک کردن تمام Sessionها	RemoveAll
ID منحصر به فرد جلسه فعلی را برمیگرداند.	SessionID
Session فعلی را خاتمه میدهد. اگر کاربر پس از دستور فوق درخواست یک صفحه جدید کند به عنوان کاربر جدید در نظر گرفته می‌شود.	Abandon
تغییر مهلت پیش فرض ختم جلسه. این خصوصیت هر عددی که باشد بعد از همان قدر دقیقه اگر کاربر درخواستی به سرور نفرستد Session ختم می‌شود.	TimeOut

نکته: از طریق فایل web.config نیز می‌توان مهلت ختم جلسه را تغییر داد:

```
<configuration>
  <system.web>
    <sessionstate timeout="60" />
  </system.web>
</configuration>
```

Event ها یا وقایع جلسه ها دو مورد هستند: Session_Start و Session_End. که Session_Start وقتی رخ می دهد که جلسه آغاز و Session_End وقتی رخ می دهد که جلسه خاتمه پیدا کند. این Event ها را باید در فایل Global.asax تعریف کرد.

در زیر یک مثال عملی از این رویدادها را خواهید دید:

```
<html>
  <head>
    <title>SessionCount.aspx</title>
    <Script Runat="Server">
      Sub Page_Load()
        lblSessionCount.Text = Application("SessionCount")
      End Sub
    </Script>
  </head>

  <body>

    Current Sessions:
    <asp:Label ID="lblSessionCount" Runat="Server" />

  </body>
</html>
```

Default.aspx

```
<Script Runat="Server">
  Sub Session_Start()
    If Application("SessionCount") Is Nothing Then
      Application("SessionCount") = 0
    End If

    Application("SessionCount") += 1
  End Sub

  Sub Session_End()
    Application("SessionCount") -= 1
  End Sub
</Script>
```

Global.asax

بطور کلی برای نگهداری مقادیر Sessionها در ASP.NET سه روش وجود دارد: درون پروسه (In Process)، ذخیره در سرویس ویندوز و ذخیره در SQL Server.

Sessionها به طور پیش فرض در داخل پروسه مدیریت می شود و تمام آیتمهایی که در Sessionها می سازیم در همان پروسه وب سرور ذخیره می شوند. مهمترین مشکل این روش این است که اگر به هر دلیل سرور از کار بیفتد و یا Web Application ما دستکاری شود، تمام داده ها از بین میرود و از طرف دیگر بسط پذیری را در سایت محدود می کند و نمی توان آن را به اشتراک گذاشت.

اما با استفاده از تکنیک ذخیره در پایگاه داده SQL Server می توان حتی در صورت از کار افتادن سرور نیز اطلاعات را حفظ کرد. تعریف اشیای ضروری در SQL Server به منظور مدیریت داده های جلسه با اجرای بچ فایل InstallSqlState.sql صورت می گیرد. بعد از این کار باید فایل web.config را نیز به شکل زیر تغییر داد:

```
<configuration>
  <system.web>
    <sessionstate
      mode="SqlServer"
      sqlConnectionString="Server=127.0.0.1;UID=sa;Pwd=YourPassword" />
    </system.web>
  </configuration>
```

باز کردن پنجره Popup در ASP.NET با استفاده از Code Behind

قطعا در بسیاری از Application های شما مواردی پیش آمده است که شما نیاز داشتید صفحه های خاص را در یک پنجره Popup باز کنید. این کار به سادگی با استفاده از یک خط کدنویسی ساده جاوا اسکریپت امکان پذیر است. حتی در صورتی که فایلی که کد جاوا اسکریپت شما در آن قرار می گیرد یک فایل Server Side باشد نیز به علت اینکه در تمامی زبانهای Server Side پیش از ASP.NET کلیه کدها به صورت Block Script و لابلای کدهای HTML قرار می گرفت منعی در استفاده از بلاکهای جاوا اسکریپت نیز وجود نداشت. با ظهور ASP.NET و قابلیت قدرتمند آن یعنی Code Behind برنامه نویسان از کلنجار رفتن با کدهای تو در تو طولانی رها شده و کلیه کدهای Server Side خود را در Code Behind می نویسند و حاصل کار آنها یک فایل .aspx است که در آن اثری از کدنویسی غیر از تگهای HTML (و تگهای توسعه یافته ASP.NET) دیده نمی شود. حال اگر شما بخواهید یک Popup باز کنید چه می کنید؟ آیا تابع جاوا اسکریپت بازکننده Popup را لابلای کدهای HTML صفحه خود می نویسید؟ این روش به نظر صحیح نمی آید. چرا که ما خود را از قابلیت خوانانویسی کدهای HTML محروم کرده و باز قسمتی از برنامه نویسی خود را به درون فایل اصلی خود انتقال داده ایم. چه می توان کرد؟ آیا راهی برای اینکه ما با استفاده از کدنویسی در Code Behind یک Popup باز کنیم هست؟ پاسخ مثبت است. در ادامه مقاله به این موضوع می پردازیم: در این روش در واقع ما کدهای جاوا اسکریپت خود را به صورت یک attribute به کنترل مورد نظر خود Add می کنیم. اما برای اینکه برای هر کنترل کدنویسی را تکرار نکنیم این قسمت را درون یک ساب روتین قرار می دهیم:

```
using System.Web.UI.WebControls;

public void OpenPopUp(WebControl opener, string PagePath) {

    string clientScript;
    clientScript = "window.open('" + PagePath + "')";
    opener.Attributes.Add("OnClick", clientScript);
}
```

طرز کار بسیار ساده است! مقادیر ورودی یک کنترل و صفحه ای است که باید باز شود. کد جاوا اسکریپت در یک متغیر رشته ای به نام clientString قرار داده می شود و با استفاده از متد Add به کنترل اضافه می شود. متد Add دو مقدار دریافت می کند: ابتدا نام property و یا event و سپس مقدار آن که پس از Generate شدن صفحه به تگ HTML تولید شده اضافه می شود.

حال شما می توانید به راحتی به صورت زیر از این روتین استفاده کنید:

```
OpenPopup(Button1, "http://www.google.com");
```

که در آن Button1 یک کنترل از نوع asp:button است. طرز کار بدین صورت است که پنجره بلافاصله پس از فراخوانی روتین باز نمی شود بلکه کنترل شما را برای رویداد مورد نظر تنظیم می کند و در هنگام وقوع رویداد (مثلا کلیک کردن روی دکمه) روتین اجرا می شود.

با کمی کدنویسی اضافه می توان ظاهر پنجره باز شده را نیز کنترل کرد. برای مثال شما می توانید طول و عرض پنجره، عنوان آن، اسکروول بار و... را تنظیم کنید:

```
public shared void OpenPopUp(WebControl opener, string PagePath,
string windowName, int width, int height) {
```

```

string windowAttribs;
string clientScript ;

windowAttribs = "width=" + width + "px," + "height=" + height + "px,"
    "left='"+((screen.width - " & width & ") / 2)+''," +
    "top='"+ (screen.height - " & height & ") / 2+"'";

clientScript = "window.open('" + PagePath + "'," + windowName + "'," +
    windowAttribs + "'); return false;";

opener.Attributes.Add("OnClick", clientScript);
}

```

مزیت استفاده از این روش این است که به علت اینکه شما این تابع را یک بار و آن هم در کلاس و یا Code Behind قرار می‌دهید می‌توانید به راحتی خطاهای برنامه نویسی خود را قبل از اینکه اجرای کد خود را آغاز نمایید، پیدا کنید.

آشنائی با فرم‌های وب در ASP.NET

به عنوان یک برنامه نویس ویژوال بیسیک شما می توانید برنامه های تحت اینترنت نیز بنویسید. به طور معمول برنامه نویسان ویژوال بیسیک به سمت ASP که یک تکنولوژی از مایکروسافت است متمایل هستند. دلیل این امر هم شباهت میان VB و VBScript می باشد. بزرگترین ایرادی که ASP کلاسیک دارد نداشتن یک محیط ویژوال مانند فرم‌های معمولی بیسیک است. مایکروسافت با Visual InterDev سعی کرد این کمبود را جبران کند اما چندان موفق نبود. بالاخره در ویژوال بیسیک دات نت ترکیبی از InterDev و ویژوال بیسیک وجود دارد و امکانات فرم‌های ویژوال بیسیک را برای اینترنت نیز فراهم می کند.

Web Form ها یکی از اجزای تکنولوژی ASP.NET است که به برنامه نویس های اکثر زبان ها این امکان را می دهد که یک قالب ویژوال با HTML و یک محیط برنامه‌نویسی تحت سرور با کدهای پیشرفته داشته باشند.

Web Form ها در عمل

بهترین راه برای فراگیری این تکنولوژی یک مثال عملی از آن است. پس از مثال معروف Hello World برای شروع استفاده می کنیم.

آماده سازی محیط

قبل از شروع ابتدا باید نرم‌افزارهای مورد نیاز را از روی لیست زیر نصب کنید. اگر ویژوال استودیو دات نت را به شکل کامل و بر روی ویندوز ۲۰۰۰ یا اکس پی نصب کردید احتیاج به مراحل زیر ندارید.

سیستم عامل شما باید حتماً از نوع ان تی باشد، ویندوز ۲۰۰۰ (سرور یا Professional)، ویندوز اکس پی Professional و یا ان تی سرور ۴. باید NET Framework. بر روی سروری که می خواهید با آن کار انجام دهید یا برنامه شما بر روی آن اجرا خواهد شد نصب شده باشد. اگر ویژوال استودیو دات نت را نصب کرده اید مشکلی در این مرحله ندارید. مایکروسافت توصیه کرده که سیستم فایل هارد دیسک سرور شما بهتر است NTFS باشد، هم به دلیل مسائل امنیتی و هم سرعت بیشتر.

Hello World

در فرم مخصوص ایجاد یک پروژه جدید ASP.NET Web Application را انتخاب کنید و نام آن را HelloWorld قرار دهید. دقت کنید که مکان ذخیره پروژه <http://localhost> باشد.

سپس بر روی کلید OK کلیک کنید تا یک Solution جدید ایجاد گردد. به طور قراردادی ویژوال استودیو یک Web Form با نام WebForm1.aspx ایجاد می کند. دقت کنید که پسوند فایل چه تغییری کرده است.

وقتی بر روی کلید OK کلیک می کنید چند عمل در پشت صحنه انجام می شود. به غیر از ایجاد کردن یک شاخه در دایرکتوری Visual Studio Projects، ویژوال استودیو یک Web Application نیز در سروری که انتخاب کرده اید ایجاد می کند. بر روی سرور، ویژوال استودیو دات نت:

یک دایرکتوری با نام پروژه در شاخه inetpub/wwwroot ایجاد می کند. این دایرکتوری را به عنوان یک IIS Application معرفی کرده و اجازه اجرای Script را بر روی آن می دهد.

اگر FrontPage Server Extensions را نصب کرده باشید یک FrontPage Web ایجاد می کند تا با FrontPage هم بتوانید به آن دسترسی داشته باشید.

می توانید همانگونه که با فرم های معمولی ویژوال بیسیک کار می کردید از Web Form ی که جلوی شما است استفاده کنید، یعنی به شما امکان استفاده از Toolbox و استفاده از کامپوننت های درون آن بر روی Web Form داده شده است. یک Label را از Toolbox برداشته و بر روی قسمت بالای فرم قرار دهید و خاصیت Text آن را به Hello World تغییر دهید.

برای این مثال تمام کاری که لازم بود انجام شود را انجام دادیم. حالا می توانیم برنامه را اجرا کنیم. قبل از اینکار از Toolbar و در قسمت Solution Configuration به جای Release،Debug را انتخاب کنید. حالا بوسیله کلید F5 برنامه را اجرا کنید. اگر هیچ مشکلی در سیستم نباشد باید صفحه مرورگر باز شود و فایل WebForm1.aspx نمایش داده و بر روی آن Hello World نوشته شود.

بر روی صفحه مرورگر کلید سمت راست موس را بزنید و View Source را انتخاب کنید تا ببینید چه مطالبی در سورس این صفحه آمده است. همانطور که می بینید کدهایی به HTML است که بوسیله فایل aspx در زمان اجرا ایجاد شده است.

همانطور که می بینید یک HTML Form در این متن دیده می شود در حالی که ما چنین چیزی را اضافه نکرده بودیم، درباره این مساله در ادامه توضیح خواهیم داد. Label ی که اضافه کرده بودیم در تگ Span قرار دارد. تگ Span مانند یک Container برای Label ما است و اطلاعات آن را در خود نگهداری می کند. به ویژوال استودیو دات نت باز می گردیم.

همانطور که دیدید Web Form ها خیلی شبیه فرم های معمولی ویندوز هستند. در Web Form جدیدی که می سازیم این خاصیت را بیشتر امتحان می کنیم. در برنامه Hello World که ایجاد کردیم تنها یک Web Form داشتیم: WebForm1.aspx. یک Web Form دیگر می سازیم تا کارهای بیشتری با آن انجام دهیم.

منوی Project | Add Web Form را انتخاب کنید. در فرمی که باز می شود Web Form را انتخاب کنید و مطمئن شوید که نام آن WebForm2.aspx است. (قبل از این کار ویژوال استودیو را از حالت اجرای برنامه خارج کنید)

بر روی Open کلیک کنید تا WebForm2.aspx در Solution ایجاد شود. بر روی WebForm2.aspx در Solution Explorer دو بار کلیک کنید تا مطمئن باشید که فرمی که تازه ایجاد کرده اید فعال است. مانند مثال قبلی یک Label بر روی فرم قرار دهید، سپس یک Button در زیر آن قرار دهید و اندازه هر دو را یکسان کنید. بر روی Label کلیک کنید و از پنجره Properties خاصیت ID را انتخاب کنید و آن را به lblText تغییر دهید. سپس بر روی کلید کلیک کنید و ID آن را به btnSubmit تغییر دهید. بر روی کلید یک بار کلیک کنید، سپس کلید Enter را بزنید تا به قسمت نوشتن کد برای این کلید وارد شوید.

در ASP.NET هر کدام از کنترل ها، کدی در پشت صحنه برای خود دارند. همانطور که مشاهده می کنید یک روال با نام btnSubmit_Click وجود دارد که هنگامی که بر روی کلید کلیک می شود اجرا می شود. کدی که در این روال نوشته شده باشد در سرور اجرا می شود و نه در مرورگر کامپیوتر کاربر. کد زیر را در روال مورد بحث بنویسید:

```
lblText.Text = "Hello World"
```

همانطور که مشاهده کردید IntelliSense وارد عمل شده و وقتی بعد از lblText، نقطه را تایپ کردید لیستی از خواص و متد های مربوط به Label را به شما نمایش داد. این خاصیت در

InterDev هم وجود دارد ولی در ویژوال استودیو دات نت از امکانات بیشتر و لیست پرمحتواری برخوردار است.

پنجره کد را ببندید و به قسمت طراحی Web Form بروید، خاصیت Text کلید را به Submit تغییر دهید. حالا برنامه را امتحان می کنیم. اگر سعی کنید تا برنامه را بوسیله کلید F5 اجرا کنید دوباره WebForm1.aspx نمایش داده خواهد شد، زیرا که این فرم، فرم ابتدایی در پروژه ما است. برای اینکه WebForm2.aspx به فرم ابتدایی تبدیل شود در پنجره Solution Explorer بر روی WebForm2.aspx کلید سمت راست موس را بزنید و سپس Set As Start Page را انتخاب کنید. حالا می توانید برنامه اجرا کنید.

Web Form جدید، WebForm2.aspx در مرورگر اینترنت نمایش داده می شود در حالی که بر روی آن یک Label و یک کلید وجود دارد. بر روی کلید کلیک کنید تا متنی که تایپ کرده بودید در Label نمایش داده شود. همانطور که می بینید برنامه مانند فرم های معمولی ویندوز اجرا می شود.

