Web Programciligina Giris

Önce basitçe dosya sunuculari vardi. Internet browserlari bir arayüz, sunuculardan aldigi dosyayi kullanicinin ekraninda gösterirlerdi. Ama her seyin hareket kazandigi günümüzde, Internet'in sabit kalmasi imkansizdi. Düsünüldü, tasinildi ve Ortak Geçit Arayüzü (CGI) yaratildi.

Ilk sunucular sadece harddisklerdi aslinda. Kullanici verilen IP'deki "harddisk"e baglanir. Ordan uzantisi HTM olan bir dosya indirirdi. HTM dosyasi kullanicilarin browserlari tarafindan derlenir, içinde <img> taglerindeki dosyalar da indirilir, ekranda bir Word belgesi gibi sekiller yerine konulur ve gösterilirdi.

O zamanlarda, günümüzde kullandigimiz Frontpage, Dreamweaver (hatta MS Word 97 ve sonrasi) gibi kolay tasarim araçlari bulunmadigi için, sayfalari güncellemek, hatalari kontrol etmek gibi görevler ancak çok dikkatli insanlarin, özenli tasarimcilarin oldukça fazla zaman alan ve almasi da gereken

islerdi. Eger ne kadar zaman aldigini ögrenmek istiyorsaniz, herhangi bir portal sitenin ana sayfasini kaydedip, HTML kodunu Not Defteri veya herhangi bir metin editörü ile açip, sadece 1-2 haber kismini resimleri ile birlikte degistirmeye çalisabilirsiniz. Eger bir de sitenizin 100'den fazla güncellenecek sayfasi varsa durum oldukça kötü anlamina gelmektedir.

Bunun böyle sürmeyecegini öngören programcilar, web sunucularina ve dogal olarak Internet sayfalarina biraz daha dinamizm ve hiz kazandirmak için yöntem arayisina girdiler. Ilk web sunuculari Unix tabanli sistemlerde çalisiyordu ve yine Unix tabanli sistemlerin dogal dili olan C ile yeni bir sistem olusturma çabalarina girildi. Mantik basitti. Kullanicinin browseri sunucudan bir HTM dosyasi istediginde, arkaplanda aslinda normalde sunucu ekranindan dahi çalistirilabilecek bir dosyayi çalistirip, dosyanin çiktisini HTML olarak geri gönderecekti. Kullanici, sunucunun ne yaptigindan haberi olmadan ekraninda sadece "yaratilmis" HTML dosyasini görüntüleyecekti. Kullanicilar, bir nevi o sunucudaki dosyayi kontrol edebildigi için buna Ortak Geçit Arayüzü (Common Gateway Interface veya kisaca CGI) denildi.

Ilk basta hersey çok yolunda gidiyordu. Kullanicilar dinamik sayfalari gördükleri için mutlu, sunucu sahipleri sitelerini daha hizli güncelleyebildikleri için memnundu. Ta ki Internet kullanicilarinin sayilari aniden patlama yasayana dek.

CGI'lar sonuçta, basit, harici programlardi. Kullanici girdiginde çalistirilir ve isi bitince kapatilirdi (kill process). Kullanici sayisinin artmasi burada devreye girdi. Ardi ardina siteye giren kullanicilar, ayni programi çalistirinca, sistemin yavaslamasi ile birlikte, bir süre sonra sunucu bu sürece yetisememeye, sonrasinda sunucunun islemez hale kadar gelebilmesine neden olabiliyordu. Daha sonra PERL gibi bir sistem gelistirilse de, CGI'ya göre performansi yine tartisilirdi.

Bunun üzerine her ne kadar benzeri bir sunucuyu Windows NT 4.0 Server üzerine eklemis olsa da, Microsoft'u yeniden harekete geçirmeye zorladi. JavaScript'in tüm browserlarda standart haline gelmesi, kendi yarattiklari VisualBasic Scripting dilini en azindan sunucularda kullanilabilirdi.

Bunun üzerine IIS (Internet Information Services) 3.0 versiyonun

üzerine ASP (Active Server Pages) adi altinda bir web programlama dili gelistirdiler. CGI ve PERL'in aksine, ASP'nin belli basli 3 önemli özelligi bulunmaktaydi.

1. Belirli bir programlama diline bagli degildi. Sunucu makinasinda desteklenen her türlü programlama dilinin kullanabiliyordu.
2. Tüm kullanicilarin ayni programi ayni kullanici gibi kullanilmasi degil, sanki her kullanici bir Windows oturumu açiyor gibi kullanmasiydi.
3. Component (Dis bilesenler) kullanilabiliyordu.

IIS 4.0 (NT Option Pack ile birlikte gelmektedir) ile birlikte ASP günümüzde bulundugu konuma oldukça yaklasmis oldu.

Bunun arkasindan çiçegi burnunda Unix klonlari için de Perl'in gelistirilmis sekli olan PHP olusturuldu. ASP'nin özelliklerinin çok benzer bir kopyasi olan PHP'de de oturum açabiliyordu.

ASP' de Veritabani Kullanimi

Web programlama dillerinin en önemli özelliklerinden biri veritabanlariyla birlikte çalisabilmesidir. Peki veritabani kullanimi

bize ne avantajlar saglar?

Veritabanlari tabii ki pek çok alanda kullanilmaktadir. Ama web üzerinde görülen en yaygin kullanimi sitelerin güncel bilgilerle doldurulmasidir. Oldukça genis bir kapsama alani bulunan veritabanlari, sitelerde arama, ürün kataloglari gibi alanlarda oldukça fazla kullaniliyor. ASP de pek çok veritabanini desteklemektedir. Bunlar arasinda tabii ki Microsoft SQL server, Access veritabanlari, mySQL, Paradox gibi sisteminizde DSN kaydi bulunan bir çok veritabanini basariyla kullanabilmektedir. Bu yazi dizimizde ise en çok kullanilan veritabanlarindan biri olan Access veritabanlari üzerinde çalisacagiz. Bunun yaninda SQL server kullananlar için de örneklerimizi verecegiz.

Peki ya gelecek?

ASP'nin gelecegi simdiden belirlendi diyebiliriz. Henüz beta uygulamalariyla karsimiza çikan ve Microsoft'un .NET projesi diye adlandirdigi, C# dili üzerine kurulmus, ASP.NET simdiden oldukça basarili görünüyor. Henüz beta asamasinda oldugu için yine de kesin bir sonuca çikamasak da, süphesiz ki günümüzde kullanilan tüm sistemlerden oldukça belirgin bir sekilde daha hizli.

Web programlama dillerinde çalisma prensipleri

Yazinin basinda da belirttigimiz üzere bir web sitesinin isleyisinin en basit açiklamasi Request-Response (Istek-Yanit) diyebiliriz. Web browserinizda bir web adresi yazdiginiz anda baslamakta olan bu süreç, yine browserinizin sag alt kösesinde Bitti (Done) yazana kadar baska bir deyisle sayfanin tamami yüklenene kadar devam eder. Bu süreç içerisinde istediginiz dökümanin özelligine göre sunucu içinde bir takim islemler gerçeklesir.

Örnegin istediginiz döküman bir HTML dökümaniysa (basitçe uzantisi htm veya html ise) sunucu size hiç bir islem yapmadan o HTML dökümanini ekraninizda görüntülemeniz için "download" etmenize izin verir.

Ama web tabanli çalisan bir uygulama uzantili ise (asp, php, aspx, cgi, pl vb) sizin isteginize göre önce sunucu içerisindeki dosya, sanki siz bir program çalistiriyormusçasina derlenmeye baslanir. Çikan sonuç size HTML olarak gönderilir. Siz basitçe bir HTML sayfasi görüyormussunuz gibi sayfaya bakarsiniz.

ASP de aslinda bir metin dosyasidir. Fakat sunucuda

çalistirildiginda içinde <% %> isaretleri arasindaki kodlar sunucunun isleyecegi kodlardir ve bu kodlarin sonucu ortaya çikan sonuç ekraniniza yansayacak, siz bu kodlari browserinizdan göremeyeceksiniz.

ASP için neler gereklidir?

Kesinlikle Windows tabaninda çalisan bir web sunucusu (IIS) gerekmektedir. Linux altinda da Microsoft'a ait olmayan ASP çalistirdigini söyleyen sunucu yazilimlari yapilmis olsa da, ASP'yi verimli kullanabilmek için Windows sarttir diyebiliriz... ASP dosyalarinin üzerine çift tiklayarak çalistiramazsiniz.

Peki evde Internete girmeden ASP'mizi nasil deneyebilecegiz? Tabii ki kendi makinamiza da IIS/PWS kurarak..

Windows 95/98/ME kullanicilari:

PWS 4.0'i Option Packten bulmalisiniz. Yalniz Windows 98 kullanicilarini bir avantaji, Microsoft Windows 98 kurulum CD'sinde PWS klasörü altinda bu yazilimi bulabilirler. Fakat ne yazikki PWS Windows 98 SE (Second Edition - Ikinci Sürüm)'de bulunmamaktadir.

Windows 95 ve 98 Birinci sürüm kullanicilari ise Internet Explorer 4.0 veya daha üstü (en son sürüm 5.5) kurmak durumundalar.

Windows NT 4.0 Server - NT 4.0 Workstation kullanicilari:

Windows NT 4.0 ile IIS 3.0 gelse de ASP destegi için NT Option Pack'i download etmeliler. Tabi ki öncelikle Internet Explorer 4.01 veya daha üzerini kurmak gerekiyor.

Windows XP Proffesional, Windows 2000 Server, Windows 2000 Proffessional ve diger Windows 2000 sürümleri kullanicilari ise zaten paketlerinin içinden gelen IIS 5.0'i kullaniyorlar. Eger IIS kurulu degilse yine Denetim Masasi - Program Ekle/Kaldir - Windows Bilesenleri (Control Panel - Program Add/Remove - Windows Components)'e girip kurabilirler.

Gerekli Download Adresleri:

Internet Explorer'in son sürümü için

http://www.microsoft.com/windows/ie/default.htm

Windows NT Option Pack (veya PWS için)

http://www.microsoft.com/ntserver/nts/downloads/reco

mmended/NT4OptPk/ adresine bakabilirsiniz. (Tam kurulumu 31 Mb'dir.)

Yükleme Sihirbazi (Download Wizard) size hiç bir zorluk çikarmadan download etmenizi saglayacaktir. Önemli olan size en yakin sunucudan indirmenizdir. Geçerli sunuculardan birini seçin ve downloada baslayin. Eger hiziniz size yeterli gelmiyorsa baska sunuculardan birini deneyebilirsiniz.

ASP Objelerini Anlamak

ASP gelistiricileri için hazir gelen 6 adet ASP Objesi bulunmaktadir. Fakat ASP Objesi nedir? Bir obje, belirli method ve özellikleri olan bilesenlerin (component) bir örnegidir. Bu kösemizde bu 6 ASP objesini ve yollarini tanitacagiz. Tabi ASP gibi güçlü bir dil, sadece bu 6 objeyle sinirlanmamistir. Tabi bunlarin yaninda ADO gibi veritabani uygulamalarina giren veya CDO gibi email uygulamalarindan da bahsedecegiz. Kendi COM objelerinizi de yaratip kullanabilmeniz ile ASP, size diger hiçbir Sunucu-tarafli (Server-Side) programlama dilinin veremeyecegi bir güç kazandirmaktadir.

Alti Hazir ASP Objesi

Asagida sayacagimiz 6 obje, ASP içerisinde hazir gelen objeler olup, herhangi ekstra bir objeye gerek kalmadan kullanabileceginiz objelerdir. Siralarsak:

1. Application
2. ASPError (Yeni)
3. Request
4. Response
5. Server
6. Session

Application Objesi:

Application objesi, sunucu açildiktan sonra, sunucudan ilk istenen .asp sayfasi ile birlikte baslar ve sunucu kapanana kadar bilgiler tutulur. Application objesi ile gelen tüm degerler, sunucudan tüm kullanicilara ulasilabilir bir sekilde tutulmaktadir. Her ASP sayfasi bir sanal dizindir ve alt dizinleri Application Objesidir diyebiliriz.

ASPError Objesi:

ASP'deki hata gösterme objesidir. IIS 5.0 ile birlikte gelmektedir. Yani PWS kurduysaniz bu obje bulunmaktadir. Windows 2000 kullanicisi iseniz bu komutlar hatalari anlamakta oldukça isinize yarayacaktir.

Request Objesi:

Request objesi, bir HTTP oturumu süresince kullanicinin browserinin sunucuya aktardigi tüm degerleri depolamaktadir. Bunun içinde kullanicinin browseri, cookieler (yalnizca bulundugu alan adina ait), SSL'den geçiyorsa sertifikalar, form bilgileri vs.

Response Objesi:

Response Objesi ise Request'in tersine, kullaniciya karsi bilgi gönderilmesi için kullanilmaktadir. Bunlar sunucu degiskenler, form karsiliklari, cookieleri olabilirler.

Server Objesi:

Server objesi, sunucumuzun (IIS) bize sagladigi yollari ve özellikleri kullanmamizi saglamaktadir. Bunlar örnegin yeni obje

açimi veya sunucu bilgisayarinin içerigini görmek vs için kullanilabilir.

Session Objesi:

Session Objesi ise kullanici browserinin sayfaya ulastigi ilk an açilir ve her kullanicinin kendine özel ayarlarini barindirmaktadir. Kullanici sayfalarinizi dolasirken bu degerler korunur ve sadeceye kullaniciya özel degerleri bildirmektedir. Örnegin her kullanicinin gezindigi sayfalarda font büyüklügü veya arkaplan renginin ayri olmasini saglayabilirsiniz.

ASP Objelerinin Kullanimi

Iki çesit obje kullanimi bulunmaktadir.

Obje.yol()

Ve

Obje.özellik

Örnegin:

Ilk örnegimizde Response.Write "Bu benim Ilk ASP Sayfam"

yazmistik.

Burada Response Obje ismidir. Response Objesi sunucuda çalistirilir ve hemen arkasindaki yol (method) olan WRITE'a bakilir ve böylelikle sunucu karsi tarafa bir sonraki degiskeni kullanicinin browsera gönderecegini anlamis olacaktir.

<% Response.Write "Bu benim Ilk ASP Sayfam" %>

Objelerin yol ve özelliklerini, Obje isminden sonra bir nokta ve yol ve özellik seklinde kullanilacagini görmüs olduk. Nokta Objeyi yol veya özellikten ayirmak için kullanilmaktadir. Yukarida Response.Write'ta yol kullandigimizi söylememize ragmen belirtmis oldugumuz parantezlerin nerede kullaniliyor oldugunu merak ediyor olmalisiniz. VBScript dilinde sadece özel durumlar disinda parantez kullanma zorunlulugu kalkmaktadir. Bu da VBScript'i ve ASP'yi diger dillere göre çok daha kolay kilan özelliklerden sadece bir tanesidir.

VisualBasic bilenlerin oldukça yatkin olabilecegi bir dil olan ASP'ye basit VBScript'ing kullanarak devam edelim. Not Defteri'nde sayfamizi açalim

```
<html>
<head>
<title>Ilk ASP Sayfam</title>
<head>
<body>
<% Response.Write "Bu benim Ilk ASP Sayfam" %>
<BR>
Su anda saat : <% Response.Write Time %>
</body>
</html>
```

Kaydedip, sayfamiza browserimiz ile bakalim (http://127.0.0.1/ilkASP.asp) Farkettiyseniz su anda sayfanizda saat:dakika:saniye bazli bir saatimiz var. Unutmayin ki bu saat kullanicin degil, sunucunun saatidir. Yani ASP sayfalarini koyacaginiz hosting sirketi neredeyse, (mesela ABD) yüksek bir ihtimalle oranin yerel saatini alacaktir.

Bir kod daha ekleyelim.

<% Response.Write Time %> satirinin altina devam edersek

**
**

Su anki Tarih**: <% Response.Write Date %>**

Tekrar baktiginizda bir de tarihle karsilacaksiniz. Görmüs oldugunuz üzere VBScrip t de basitçe Ingilizce bilgisine dayanmakta.

ASP' de Komut Dizilisi

ASP'de komutlar, <% ve %> taglarinin arasina yazilir demistik. Ama iki komut ayni satira yazilamaz. Ya %> seklinde komut kapatilip tekrar <% ile baska bir komut satiri açilir veya ENTER ile bir satir asagidan devam edersiniz. Örnegin:

<% Response.Write Time %><% Response.Write Date %>

veya

<% Response.Write Time
Response.Write Date %>

gibi. Tabi sizin tercihiniz ama tekrar bir %><% yazma zorunlulugundan kurtulmaniz için 2. yolu tercih etmenizi öneririm.

Tabi yukaridaki gibi yazdigimizda aralarda Enter olmasina ragmen

sayfamiza baktigimizda yanyana durdugunu görüyoruz. Sonuçta ASP de bir HTML Yorumlayicisina gönderilir. Bu da demek oluyor ki ASP'ye tam hükmedebilmek için de iyi bir HTML bilgisine ihtiyacimiz var. Bu yüzden satir atlamasini istiyorsaniz ASP kodlarinin arasinda yine
 eklemek gerekiyor. Eger ASP Kodu içerisinden yapmak istiyorsak HTML kodlarini çift tirnak isaretleri arasina yazmamiz yeterli olmakta.

Örnegin:

```
<% Response.Write Tarih
Response.Write "<BR>"
Response.Write Date %>
```

Veya

```
<% Response.Write Tarih %><BR><% Response.Write Date %>
```

Istekler ve Yanitlar

Daha önceden HTML kursumuzu takip edenler hatirlayacaklardir. Formdaki bilgileri bir ASP veya CGI/Perl programcigina aktarildiktan sonra islemler yapilir demistik. Peki bu bilgileri ASP nasil isler? ASP konusunda Ilk objemiz Request objesi olacaktir.

Kullanimi:

Request.Method( ) seklindedir.

Request, Türkçe anlamiyla "Istek" anlamindadir. Sunucuya gelen her istek sunucunun cache'inde tutulmaktadir. Böylelikle sayfalarinizda ASP kodlama ile bu istekleri, yanitlara dönüstürebilir ve sayfalariniza degisik bir dinamizm katabilirsiniz. Geçen sayimizda daha sonra daha detayli görecegimiz Response objesinin ekrana yazi yazmak için kullanilan Write metodunu görmüstük. Formlar da sunucuya kullanicilarin girmis oldugu bilgileri gönderdigine göre, sunucunun bunu öncelikle istek olarak algilayip, ona göre karsilik vermesi dinamik sayfalarin temelinde bulunmaktadir.

Dilerseniz öncelikle sunucuya request yollayacagimiz bir HTML sayfasi yaratalim

```
<HTML>
<HEAD>
<TITLE>Request Denemesi için Formlar Sayfasi</TITLE>
</HEAD>
```

```
<BODY BGCOLOR="#CCCCCC">
<H1>Form Yollama Denemesi</H1>
<FORM METHOD="POST" ACTION="request.asp">
<TABLE CELLSPACING="2" CELLPADDING="2" WIDTH="80%"
ALIGN="CENTER">
<TR>
<TD WIDTH="20%">
Lütfen Isminizi Giriniz:
</TD>
<TD>
<INPUT TYPE="text" WIDTH="30" NAME="isim">
</TD>
</TR>
<TR>
<TD>
Lütfen Cinsiyetinizi Seçin
</TD>
<TD>
<SELECT NAME="cinsiyet" SIZE="1">
```

```
<OPTION VALUE="1" SELECTED>Erkek</OPTION>
<OPTION VALUE="2">Bayan</OPTION>
</SELECT>
</TD>
</TR>
<TR>
<TD>
ASP Ögrenmek istiyor musunuz?
</TD>
<TD>
<INPUT TYPE="radio" NAME="soru" VALUE="1"
CHECKED>Evet<BR>
<INPUT TYPE="radio" NAME="soru" VALUE="2">Hayir
</TD>
</TR>
<TR>
<TD><INPUT TYPE="submit" VALUE="Formu Gönder"
</TD>
</TR>
```

</TABLE>

</FORM>

</BODY>

</HTML>

Istekler ve Yanitlar(2. Bölüm)

Bu sayfayi InetPub\wwwroot dizininizin içinde dilediginiz bir isimde HTML veya ASP uzantisiyla kaydedebilirsiniz. Çünkü formumuzun içinde herhangi bir ASP ögesi kullanmadik.

Dikkat ettiginiz üzere Isminizi soran bir metin girdisi satiri, cinsiyetinizi soran bir seçimli menü ve ASP ögrenmek isteyip istemediginizi soran bir buton seçimi görünmekte

Formun gönderilecegi ASP dosyasinin adini ise Request.asp seklinde belirttik. Simdi ayni dizinde Yeni Metin Belgesi açalim ve ismini request.asp diye degistirelim. Metni Not Defteri ile açalim ve komutlarimizi yazmaya baslayalim:

Form'dan gelen bilgiler için Form metodunu kullanacagiz. ASP'de satir aralarina not düsmek ve ASP yorumlayicisinin (ASP.DLL) bunu es geçmesini istersek ' tek tirnak isaretini kullanabiliriz.

Basic'teki REM (Remark) komutunun yerine geçmektedir. Dim komutu ise gelecek veriler için bir veri alani açmamiza yarayacaktir. Dim komutu ile tanimladigimiz degiskenlere daha sonra yanlislikla baska bir isim vermemizi saglayacaktir. Daha detayli bilgiyi ise önümüzdeki bölümlerde görecegiz.

```
<%
' Öncelikle Formdan gelen bilgileri alalim.
Dim isim
Dim cinsiyet
Dim soru
Dim cevap

isim = Request.Form("isim")
cinsiyet = Request.Form("cinsiyet")
soru = Request.Form("soru")

%>
```

Böylelikle isim, cinsiyet, soru ve cevap gibi 4 ayri degisken açip, Bunlari Form'daki NAME'leriyle bize gelen verilere yönlendirdik.

Cinsiyet = Request.Form("cinsiyet") satirini yorumlarsak: "cinsiyet" isimli formdan gelen degeri ("VALUE") , cinsiyet diye bir degiskene ata anlaminda bir satir yazmis olduk.

<% %> isaretlerine "sinirlayici (Delimiter)" denmektedir. ASPner komutlarinin eden baslayip nereden bittigini belirtirler. En son yazdigimiz sinirlayici kaldiralim ve kodumuzu hazirlamaya devam edelim.

Simdi görecegimiz VBScript'in If..Then .. Else komutudur.

Kullanimi:

If parametre sartlari Then

Yapilacak islemler

Else

Eger sartlar saglanmiyorsa yapilacak islemler

End If

Burada sorumuza sordugumuz yaniti metne dönüstürelim. Eger soru butonlarindaki deger 1 ise bunu Evet cevabi olarak aldiralim, eger soru= "2" cevabi verilirse bunu da Hayir olarak cevap

degiskenine atayalim

```
If soru = "1" Then
Cevap = "Evet"
Else
Cevap = "Hayir"
End If
%>
```

Simdi ASP'nin HTML kismini hazirlamaya baslayalim

```
<HTML>
<HEAD>
<TITLE>Form Sonuçlari</TITLE>
</HEAD>
<BODY BGCOLOR="#CCCCCC">
<H1>Form Sonuçlari</H1>
<TABLE CELLSPACING="2" CELLPADDING="2" WIDTH="80%"
ALIGN="CENTER">
<TR>
```

```
<TD WIDTH="100%">
Isminizi <B><% Response.Write isim %></B> olarak yazdiniz
</TD>
</TR>
<TR>
<TD>
Cinsiyetinizi <B><% Response.Write cinsiyet %></B> olarak
belirttiniz.
</TD>
</TR>
<TR>
<TD>
Anket sorusuna <% Response.Write soru & " yani " & cevap %>
olarak cevap
verdiniz.
</TD>
</TR>
</TABLE>
</BODY>
```

</HTML>

Burada Response.Write metodunu degisik bir kullanimini ögrendik. Yazacagimiz degiskenleri & (ve) isaretiyle ayni komut üzerinde yazabiliyoruz.

Response.Write "PC" & " " & "Online" gibi

Ekranda PC Online olarak görünecektir.

Tabi Request objesinin kullanimi sadece formlarla da kalmiyor. Çerezler (Cookies) , Sunucu Degiskenleri (Server Variables) , Sorgu satirlari (QueryString) gibi istekleri de yine request objesi ile alabiliyoruz.

Request.ASP dosyamizi bir kez daha kopyalayarak adini reqquery.asp olarak degistirin ve not defterinizi açarak burada Request.Form olarak gördügünüz satirlari Request.QueryString olarak degistirin.

QueryString nedir?

Sörf yaparken bol bol rastladigimiz örneklerdir sorgulama satirlari. Linklerin üzerine geldiginizde bolca = ve & isareti bulunan satirlar, gidilecek dökümanin aktif bir döküman oldugunu ve bu satirlardaki degiskenlere göre degisecegini anlatir bizlere.

QueryStringler, Formlarin kullanilmasindaki GET metodundaki gibi verileri browserinizin Adres kismina biriktirip karsi dökümana öyle yollarlar. Böylelikle Formlarin her zaman kullanilamayacagi linkler gibi yerlerde kullanilmaktadir.

Yani

```
isim = Request.QueryString("isim")
cinsiyet = Request.QueryString("cinsiyet")
soru = Request.QueryString("soru")
```

ve bir browser penceresi açarak

http://localhost/reqquery.asp?isim=Adiniz&cinsiyet=Erkek&soru=1

yazip Enter'a basin. Karsiniza browserinizin adres kismina yazmis oldugunuz veriler gelmekte bunu bir çok örnekte görebilirsiniz.

Gelecek veri, ister Form'dan isterse sorgulama satirlariyla gelsin, metod kullanmadan sadece Request objesiyle bunlari almak mümkün.

isim = Request("isim")

seklinde Form veya QueryStringlerden gelen isim verisini degerlendirebilirsiniz.

VBScript'e Merhaba

Bu bölümden itibaren, ASP içerisinde en genis kullanim alani bulunan ve Visual Basic'in tüm gücünü web sayfalarimiza tasiyan oldukça basit kullanimi ile bir çok kez hayatimizi kurtaracak bir dil olan VBScript'i hizlica tanitmaya çalisacagiz.

Eger Visual Basic'e veya Basic diline asinaliginiz varsa, VBScript sizin için çocuk oyuncagidir diyebiliriz. ASP içerisinde varsayilan dil olan VBScript, Internet Explorer disindaki browserlarda istemci tarafinda çalisabilecek bir dil kabul edilmedigi için, VBScript'in istemci bazli özellikleri, MsgBox veya InputBox gibi Visual Basic

özelliklerini kullanamayacagiz fakat ASP'nin de sunucu tarafli oldugunu düsünürsek bu tür komutlari kullanmayacagimizi söyleyebiliriz.

ASP içerisinde VBScript kullanimi

ASP'nin bir çok Script diliyle çalistigini ve hangi dil ile kullaniyorsak onun servera belirtilmesi gerektigini söylemistik. Ama VBScript ASP'nin varsayilan dili oldugu için bunu belirtmek gerekmiyor. Tabi ASP içerisinde server tarafindan islenecek komutlari yazmaya baslamadan ve bitirdikten sonra da yine Delimiter (sinirlayicilar) dedigimiz <% ve %> isaretlerini koymayi unutmamak gerekiyor.

Kullandigimiz dil, sayfanin en üstünde @Language ile belirtiliyor.

<% @Language = "VBScript" %> gibi..

Eger Ingilizce biliyorsaniz, komutlari anlamak oldukça kolay olacaktir.

Ilk kodumuzu yazalim.

<% @Language = "VBScript" %>

```
<HTML>
<HEAD>
<TITLE>VBScript Örnek - 1 </TITLE>
</HEAD>
<BODY BGCOLOR="#FFFFFF">
<CENTER>
<% For I=1 to 7 %>
<H<% Response.Write I %>>VB Script </H<% Response.Write I
%>>
<% Next %>
<H7> Görmüs oldugunuz üzere Normal HTML komutlari içerisinde
de Delimiterler ile ASP kodu yerlestirebiliyoruz. </H7>
</BODY>
</HTML>
```

Degiskenler ve Degisken Yönetimi

Tüm programlama dillerinde oldugu gibi VBScript içinde de, o script içerisinde sik kullandigimiz degerleri "Degisken" adini verdigimiz tanimlara ihtiyacimiz olacaktir. Degiskenler, adindan da

anlasilacagi üzere, degerlerini sizin belirlediginiz veya programlamaniza göre VBScript'e belirttiginiz degerleri islemeniz için kullandigimiz kisa yollardir. Degiskenlerin VBScript'e önceden belirtilmesi gerekmese de, VBScript'in bu degiskenleri çok daha hizli ve dogru sekilde degerlendirmesini saglar.

Degiskenlerini tanimlamak için boyut belirleyecek komut DIM 'dir. Degisken isimleri rakamla baslamayan (fakat rakam içerebilen), Türkçe karakterler kullanamadigimiz harf ve rakamlardan olusur. Degisken isimlerinin boyutu da en fazla 255'tir. Suxxess, sayac12 vb.. Ama degiskenlere atadigimiz degerlerde böyle bir sinirlama bulunmuyor. Tek DIM komutunda, virgül ile ayirarak bir çok degiskeni tanimlayabiliriz.

```
<%

DIM sayac, dergi, isim, soyad, yas

sayac = 1

dergi = "CHIP"

isim = "Selçuk"

soyad = "Islamoglu"
```

```
yas = 24

tarih = Date()

sayi = "1"

%>
```

Böylelikle 5 tane degiskenimize degisik degerler atadik. Farketmis olacaginiz üzere sayisal degerleri tirnak isaretleri arasina almadik. Degiskenin bir oturum boyunca kullanacagi yerlerdeki davranislarini belirten Variant dedigimiz türler bulunmaktadir.

Üstteki örnege göre örnek vermek gerekirse "yas" degeri için

<% Response.Write (yas * 2) %> yazdigimizda ekrana yazilacak sonuç 48 olacaktir. Fakat tirnak içerisine aldigimiz "sayi" degiskenini 2'yle çarptigimizda script hata verecektir.. Çünkü VBScript onu bir yazi satiri (String) olarak almistir.

Veri Türleri	
Boolean (Bit)	Bu deger 1 (True - dogru) veya 0 (False-Yanlis) durumundadir.
Byte	0 ile 255 arasinda degisen sayisal degerdir.

Double	Floating Point (Kayan nokta) degerleri türüdür. 4.9E-324 ile 1.8E308 arasi degerlerde bulunabilir.
Date/Time	Tarih veya zaman bilgisini içerebilir. Bu belirlenirken sunucunun Regional Settings (Bölgesel Ayarlar) içerisinde belirtilen tarih ve zaman birimleri kullanilmaktadir.
Empty	Deger atanmamis degiskenlerdir. Isim = "" gibi..
Error	Programin hata degerleridir
Integer	Tam sayidir. Ondalik bölüm içermez. +32.768 ve - 32767 arasindaki tamsayilardir.
Long	Noktalik bölüm içermeyen (tamsayidir) fakat -2.147.482.648 ile 2.147.483.648
Null	Içinde veri bulunmayacak sekilde tanimlanmis degiskendir. Bunu empty ile karistirmamak gereklidir. Empty'de deger olarak "" vardir. Fakat null'da bir deger bulunmamaktadir.
Object	Windows OLE nesneleri kullanmak için kullanilan degiskendir.
Single	Kayan nokta degerlerinden bir digeri olan Single'in 1.4E-45 ve 3.4E38 arasinda degisen türüdür.
String	Alfanümerik karakter bütünüdür. 2 milyar'a kadar karakter

	içerebilir.

Degisken boyutlari

Diyelim ki isyerinizde çalisanlarinizin bilgileri bir çesit veritabanina kayitli ve siz bunlari degiskenlere atamak istiyorsunuz. Herbirine ayri degiskeni nasil verebilirsiniz?

Burada dizi degiskenleri veya Array yapisi ortaya çikiyor.

```
<%

DIM isciler(100)

' Burada 100 tane kaydin tek bir degiskende tutulabilecegini belirtmis olduk. Ama kayit numaralari 0'dan basladigi için 99'a kadar veriyi tek degiskene toplayabiliriz.

Isciler(0) = "Ahmet"
Isciler(1)= "Emre"
Isciler(2) = "Sahin"
....

Isciler(99) = "Mahmut"
```

%>

Böylelikle kaçinci isçinin adini yazdirmak veya islemek istiyorsak, onu yazdirabiliriz..

<% Response.Write Isciler(2) %>

yazdigimizda ekrana "Sahin" yazacaktir. Peki bu kadar veriyi girdikten sonra bir 20 isçinin daha sirketinize alindigini göz önüne alinirsa, bu verilen girilmesi için de REDIM komutunu kullaniyoruz..

Varolan degiskenimizi REDIM Isciler(120) seklinde tanimlarsak, Isciler degiskenindeki verilen silinip, yerine 120'lik yeni bir sirali degisken açilacaktir. Bunun önlemek için PRESERVE komutunu ekliyoruz..

<% Redim Preserve Isciler(120)

Isciler(101) = "Oytun"

...

%>

Böylelikle ilk 100 kaydimiz silinmeden sirali degiskenimiz

genisletmis olduk. Peki sirali degiskenler içerisine 1'den fazla veri girmek mümkün mü? Tabi ki..

```
<% DIM Isciler(120,3) %>

Isciler(0,0) = "Ahmet"

Isciler(0,1) = "Dedeoglu"

Isciler(0,2) = "Istanbul"

Isciler(1,0) = "Emre"

Isciler(1,1) = "Barkin"

Isciler(1,2) = "Ankara"

%>
```

Böylelikle tek degiskenin 120 tane kaydina ait, 3 ayri bilgiyi yine tek degiskende topladik.

Ahmet Dedeoglu'nun Istanbul'da yasadigina ait bilgiyi sadece ona ait degiskende bütünlestirmis olduk. Buna Boyut diyoruz.
Isciler(1,2) degeri , 1. kaydin 2. boyutundaki degisken anlamina gelmektedir.

Yani bu kadar kaydin arasinda sadece soyadlarini yazdirmak

istiyorsak;

```
<% For I=0 to 120
Response.Write Isciler(I,2) & "<br>"
Next
%>
```

yazmamiz yeterli olacaktir.

VB Script Operatörleri

Operatörler, bilgisayarda degiskenler arasinda mantik, aritmetik ve karsilastirma islemleri yapan ara komutlardir. Istedigimiz islemleri yaptirmak için bir çogumuzun okullarimizda ögrendigimiz mantik ve aritmetik kurallarini kullanmamiz gerekiyor.

Aritmetik Islemler:	
+	Toplama
-	Çikartma
*	Çarpma
/	Bölme

^	Üs alma
\	Tam sayi bölümü
Mod	Modüler Artimetik (Sayi düzenleri)
&	Alfanumerik toplama

Karsilastirmalar:	
=	Iki degiskenin esitligini sinar
<>	Iki degiskenin birbirine esit olmadigini sinar
>veya<	Bir degiskenin digerinden büyük veya küçük oldugunu sinar
>=veya<=	Bir degiskenin digerinden büyük veya esit, ya da küçük veya esit oldugunu sinar
Is	Nesnelerin (Object) esitligini sinar

Mantiksal Islemler:	
And	Mantikta "VE" islemidir. Iki islemin sonucu

	ayniysa sonuc dogru (True), bir tanesi bile sonucu vermiyorsa yanlis (False) degerini verir.
Or	Mantiktaki "VEYA" islemidir. Iki islemden birisinin sonucu dahi dogru olsa Dogru (True) degerini verir. Ancak ikisi de sonucu vermiyorsa yanlis (False) sonucu verir
Not	Bir ifadeyi negatif hale getirir. Baska bir deyisle, islem sonucu Dogru (True) ise bunu Yanlis (False) hale getirir.
Xor	Iki veya daha fazla islemin sonuçlarindan herhangi biri dogru ise dogru sonucu verir
Exp	Iki ayri ifadenin Denkligini sinar.
Imp	Iki ifadenin degerlerinin mantiksal çarpimini sonuç olarak verir.

Operatörler aritmetik islem sonuçlarini hazirlarken, matematiksel formüllerde kullandigimiz gibi bir öncelik sirasi kullanirlar. Islemci, öncelikle üs alma islemleri, sonrasinda çarpma ve bölme ve en son olarak toplama ve çikarma islemlerini yapar. Eger

islemlerimizde öncelikle + veya çikarma gibi islemler varsa bu islemleri parantezlerle kapayarak halledebiliriz.

Örnegin: 86-3*2 'nin sonucu 80'dir. Yani öncelikle çarpma islemi yapilir ve ilk sayidan çikarilir. Fakat (86-3)*2 'nin sonucu 166'dir.

Mantiksal Kontroller

VBScript'te veya ASP sayfamizda belirli sartlara göre sayfalar veya programlar yaratilir. ASP'nin dinamik sayfalar olmasinin sebebi de budur. Ama bu sartlari saglamak ancak mantiksal kontrollere göre yapilabilir.

If..Then..Else..

Basitçe:

If sart(lar) Then (Eger sartlar oluyorsa)

Belli bir uygulamayi yap

Else (Sartlar uymuyorsa)

Baska bir uygulamayi yap

End If (Eger'i kapat)

seklinde kullanilir. Önceleri internetimiz.com üzerinde kullandigimiz basitçe bir örnegi verelim. Türkiye saatine göre size

Hosgeldiniz, Iyi aksamlar gibi mesajlar veriyordu. Bunun benzeri bir kodu uygulamaya geçirelim.

```
<HTML>
<HEAD>
<TITLE>Saate göre mesaj</TITLE>
</HEAD>
<BODY BGCOLOR="#FFCC00">
<%
Dim Mesaj

If Hour(Now) >= 5 AND Hour(Now) <= 11 Then
Mesaj = "Günaydin"
ElseIf Hour(Now) >= 12 AND Hour(Now) <= 18 Then
Mesaj = "Hosgeldiniz"
ElseIf Hour(Now) >= 19 AND Hour(Now) <= 22 Then
Mesaj = "Iyi aksamlar"
Else
Mesaj = "Iyi geceler"
End If
%>
```

```
<CENTER>
<H2>Saat su anda: <% Response.Write Time %></H2>
<H1><% Response.Write Mesaj %></H1>
</BODY>
</HTML>
```

Burada bir de Elseif 'i görüyoruz. Birden fazla durum söz konusu ile kullanilan bu komutla diger durumlara da gönderebiliyoruz. End If ile duruma ait uygulamanin sonlandigini görüyoruz. Ayrica Hour ve Now() gibi ASP'nin kendi fonksiyonlarini da kullanmis olduk. Yeri gelmisken zaman fonksiyonlarini da kisaca özetleyelim.

Date()	O anki Tarihi verir (24.07.2001 gibi)
Now():	O anki Tarih ve saati verir (24.07.2001 14:28:31
Day(Tarih) :	Verilen tarihteki gün kismini bulur Day(Now()) sonuç: 24
Month(Tarih) :	Verilen tarihteki ay kismini bulur Month(Now()) sonuç: 7
Year(Tarih) :	Verilen tarihteki yili verir Year(Now()) sonuç:

	2001
Hour() :	Saati verir
Minute() :	Dakikayi verir
Second() :	Saniyeyi verir
Weekday():	Regional Settings (Bölgesel Ayarlar) özelliklerine göre fonksiyon içerisine yazdiginiz tarihin haftanin kaçinci günü oldugunu söyler. Örnegin çikan sonuç 1 ise Bölgesel Ayarlari Türkçe olan bir makinada Pazartesi anlamina gelir.
WeekdayName():	Verdiginiz tarihin gün ismini verir. Örnegin WeekdayName(Weekday(Now()))

If .. Then..Else kalibindan baska bir yol da SELECT CASE yoludur. Belirli bir duruma göre yapilacak islemler CASE'ler yardimiyla siralanip, VBScript'in bunlari islemesi saglanir.

Kullanimi:

```
SELECT CASE degisken
CASE degisken degeri1
.. 1. durumda yapilacak islemler...
```

```
CASE degisken degeri2
.. 2. durumda yapilacak islemler
CASE degisken deger3
...3. durumda yapilacak islemler
...
...
END SELECT
```

Az önce uyguladigimiz örnegi, SELECT CASE ile yaptigimizda içerisindeki ASP kodumuz su sekilde olacaktir.

```
<%
SELECT CASE Hour(Now)
CASE 5,6,7,8,9,10,11
Mesaj = "Günaydin"
CASE 12,13,14,15,16,17,18
Mesaj = "Hosgeldiniz"
CASE 19,20,21,22
Mesaj = "Iyi aksamlar"
CASE ELSE
```

```
Mesaj = "Iyi geceler"
END SELECT
%>
```

Burada "Hour(Now) fonksiyonunun aldigi degerlere göre asagidaki durumlari uygula" komutu olan SELECT CASE Hour(Now) 'dan sonra CASE'lerle olusabilecek durumlari yazdik ve CASE ELSE ise daha önce belirtilen CASE durumlari disinda herhangi bir durum oldugunda kullanilacak durumlari belirtmenize yarar.

VScript Döngüleri

ASP vede VBScript'te duruma göre degiskenleri ve olasi durumlari kontrolde bir diger durum ise döngülerdir. Bir islemi belirli sartlara gelinceye kadar veya belirli sartlari oldugu sürece döngüye sokmak, elimizdeki çok sayida verinin islenmesine olanak saglar veya sartlarin devanmini getirir. Daha önceki programlarimizda da kullandigimiz 2 tür döngü bulunmaktadir.

For .. Next Döngüsü

For döngüsü belirli bir sarttan baska bir sarta giderken arasina yazdigimiz fonksiyonu çalistirmamiza yaramaktadir. For

döngüsünü bir sayaç gibi kullanilmaktadir. Bu sayacin çesidi yine size kalmis.

"Degisken Yönetimi" basligi altinda verdigimiz örnegi tekrar ele alirsak;

```
<% For I=0 to 120
Response.Write Isciler(I,2) & "<br>"
Next
%>
```

0'dan 120'ye kadar I degerini birer artirarak Isciler'in soyadlarini yaz anlamina gelen yukaridaki örnekte adim belirtmedigimiz için 1'er 1'er sayi arttirilarak gidiliyor.

For .. Next döngüsünün kullanimini tam olarak yazmak gerekirse

```
<% FOR degisken=baslangic TO bitis STEP adim sayisi
```

Islemler

```
NEXT %>
```

Olarak verebiliriz. Örnegin 120 sayisinda 2'ser 2'ser geri dogru 0

sayisina ulasmak için yazabilecegimiz komut <% For i=120 to 0 STEP (-2) %> olacaktir. Madem böyle bir döngü yaratiyoruz "i" degiskenini asagidaki döngüde kullanalim.

```
<HTML>
<HEAD>
<TITLE>Döngüler</TITLE>
<HEAD>
<BODY BGCOLOR="#FFCC00">
<% For i=1 to 10
Response.Write "Su anda For döngüsü " & i & ". Kez çalistirildi." &
"<BR>"
Next
%>
</BODY>
</HTML>
```

Böylelike 2 sinir deger arasinda istedigim adimda ekrana yazi yazdirmis olduk.

While .. Wend Döngüsü

Kullanimi " Sart saglaniyor iken" ...durumu uygula ..." Tekrar dene .." durumudur..

Örnegin:

```
<%
sayac = 0
While sayac <= 100
Response.write sayac & "<br>"
Sayac = sayac + 1
Wend
%>
```

Bu örnekte sayac 100'den küçük veya esit oldugu süre içerisinde While ve Wend arasindaki kodu tekrar etmesini istedik. Fakat kodun içinde gördügünüz üzere sayac'i da 1 arttirmayi unutmadik. Aksi takdirde farkedeceginiz üzere sonu olmayan bir tekrar ve makinanizin kilitlenmesi sizi bekleyebilir.

VScript Döngüleri (Bölüm-2)

Do ... Loop Döngüsü

Iki türlü yapilis biçimi olan "Yap".."Döndür" komutu olan Do.. Loop, While Wend'i de içine katan bir yapisi bulunmaktadir..

Do while sart

Yapilacak Islemler

Loop

Burada "Sart dogru oldugu sürece aradaki islemleri yapip tekrar et" anlaminda bir döngü kullandik. Do ve Loop'un diger kullanimina gelirsek

Do until sart

Yapilacak islemler

Loop

Bu kullanimda ise "Sart gerçeklesinceye kadar aradaki islemleri yapip tekrar et" döngüsünü görüyoruz. Bir örnekle pekistirelim

<HTML>

<HEAD>

<TITLE>ASP ile ZAR ATMAK</TITLE>

</HEAD>

```
<BODY BGCOLOR="#FFCC00">
<%
if Request.Form("geleceksayi") <> "" Then
Dim GelmesiIstenenSayi, Gelen, Sayac
GelmesiIstenenSayi = Int(Request.Form("geleceksayi"))
Gelen = 0
Sayac = 0
Randomize
Do while Gelen <> GelmesiIstenenSayi

Sayac = Sayac + 1
Gelen = Int(Rnd * 6) + 1
Response.Write "<b><H4>" & Sayac & ". Atista gelen zar:</b> "
& Gelen & "</H4><br>"
Loop

Response.Write "Istediginiz zarin gelmesi için, zar " & Sayac & "
kez atildi." & "<br>"
```

```
Response.Write "Bir daha deneyin.."

End If

%>

<H2>Lütfen Atmak istediginiz zari seçiniz</H2>

<FORM METHOD="POST" ACTION="<%=

Request.ServerVariables("SCRIPT_NAME") %>">

<SELECT SIZE=1 NAME="geleceksayi">

<% For i=1 to 6 %>

<OPTION VALUE=<%= i %>><%= i %></OPTION>

<% Next %>

</SELECT>

<INPUT TYPE="Submit" NAME="Gonder" VALUE="Zar At">

</FORM>

</BODY>

</HTML>
```

VBScript'in Derinlikleri

VBScript'in ince ayrintilari ASP sayfalarimizda çok daha önem kazaniyor.

Geçen sayimizda döngülerden bahsetmistik, For.. Next, Do.. Loop gibi. Döngülerin genel olarak karsimiza çokça çikan sorunlarda bize oldukça faydali oldugundan bahsetmistik. Bu sayimizda ise VBScript'in bize oldukça zaman kazandiracak olan diger fonksiyonlarindan bahsedecegiz. Bu fonksiyonlari operatörler ve sorgularla nasil kullanabilecegimizi görecegiz.

Matematiksel Fonksiyonlar

Abs(degisken)

Verilen degerin mutlak degerini alir. Yani degerimiz 1.456 ise Abs(1.456) 1 degerini verir.

Atn(degisken)

Verilen degerin Atanjantini alir

Cint(Deger)

Verilen degeri tam sayiya yuvarlar veya çevirir.

Cos(Deger)

Degerin kosinüsünü alir

CDbl(Deger)

Double degere dönüstür. Yani negatif sayilari için -1.79769313486232E308 ile -4.94065645841247E-324 arasi; pozitif degerler için 4.94065645841247E-324 ila 1.79769313486232E308 arasi degerlere çevirir.

CSng(Deger)

Degerin Single degerine çevirir. Yani -3.402823E38 ile -1.401298E-45 arasi veya 1.401298E-45 ile 3.402823E38 arasi degerlere dönüstürür.

Sgn(Deger)

Verilen degerin isaretini verir. Yani sayi negatifse -1, pozitifse 1, 0 ise 0 sonucunu verir.

Sin(Deger)

Verilen degerinin sinüsünü verir.

Sqr(Deger)

Verilen degerin kare kökünü gösterir.

Tan(Deger)

Verilen degerin tanjantini alir.

String Fonksiyonlari

Visual Basic'te string tanimlamasi, karakter veya sari dizisi olarak söylenebilir. Stringlere bir çok örnek verilebilir. "Benim adim Emre" veya "Ahmet saat 3:00'te buraya gelecek" birer string yapisidir. Sayfalarda gerek formlar, gerek querystringlerle tasinan bilgiler de string örnegi olarak verilebilir. Ama sayfalarimiza bu verileri isleyebilmek için, onlari isleyebilecek fonksiyonlara ihtiyaciniz olacaktir.

InStr

Karakter dizilerinin içinde baska bir karakter dizisini veya karakteri sorgulamaya yarayan bir komut olan InStr, bulunup bulunmadigini, bulunursa kaçinci karakter veya baslama noktasindan itibaren kaçinci karakterde basladigina dair bilgileri verir.

Örnegin

<%

Dizi = "ASP kursu içerisinde VBScripti isliyoruz"

```
AranacakStr = "VB"

Kacinci = InStr(Dizi, AranacakStr)

Response.Write Kacinci

%>
```

Burada ekranimizda görünecek olan sayi 22'dir. "VB" yazisi Dizi degiskeninde 22. karakterden baslamaktadir. InStr'nin bir diger kullanimi ise, hangi karakterden sonra baslayacagimizdir.

Kacinci = InStr(3, Dizi, AranacakStr)

Yani bastan 3. karakterden sonra "VB" karakter dizini kaçinci karakterde gelmektedir seklinde bir sorgu yaratmakta ve sonucunu 19 olarak almaktayiz.

InStr, karakter karsilastirma olarak da kullanabilecegimiz bir komuttur. Örnegin formunuzda, kullanicilarinizda email adresi talep ettiniz ve email adresinin en azindan normlara uygun olmasini istiyorsunuz. Formumuzu yazalim

Form.asp

```
<HTML>
<HEAD>
<TITLE>Email Adresinizi giriniz</TITLE>
</HEAD>
<BODY BGCOLOR="#FFCC00">
<FORM METHOD="POST" ACTION="emailkontrol.asp">
<b>Lütfen Emailinizi Giriniz: </b><br>
<INPUT NAME="email" TYPE="TEXT" SIZE="30">
</FORM>
</HTML>
```

Burada gönderecegimiz ASP'nin de kodunu birlikte yazalim

Emailkontrol.asp

```
<%
Dim email
Email = Request.Form("email")
' Emaillerde standart olarak bulunmasi gereken veriler @ isareti
ve en az 1 tane noktadir.
```

```
Kontrol1 = InStr(Email, "@")

Kontrol2 = InStr(Email, ".")

' Kontrol1 ve kontrol2'nin sonucunun 1'den büyük olmasi

gerekmektedir. If ile sorgulamamizi yapalim

If Kontrol1 > 0 AND Kontrol2 > 0 Then

Response.Write "Email dogrulandi"

Else

Response.Write "Emailiniz hatalidir"

End If

%>
```

VBScript'in Derinlikleri (Bölüm-3)

Len(KarakterDizisi)

Bir karakter dizisinin kaç harf veya karakterden olustugunu vermektedir.

Örnegin:

```
<% Dizi = "Emre"

Response.Write Len(Dizi)

%>
```

Ekrana 4 olarak yansiyacaktir. Üstteki email örnegimizde de bunu kullanabiliriz.

Email adresleri isim@domain.tür veya isim@domain.tür.ülke seklindedir.

@ , (.) nokta ve bunlarin kullanici ismi için en az 1, domain için en az 1 ve domain.tür (com.tr, co.uk, com, net, org) gibi uzantilarda da en az 3 harf olacagi için 7 karakterden kisa olmamalidir.

Üstteki örnegimize emailkontrol.asp dosyamizdaki If satirini söyle de degistirebiliriz.

```
If Kontrol1 > 0 AND Kontrol2 > 0 AND Len(Email) >= 7 Then
```

Yani Emailimizin uzunlugunun da 7 karakterden büyük olmasi sartini eklemis olduk.

Left(Degisken, Sayi): Bir yazi dizisindeki karakterlerin, soldan "Sayi" kadarini gösterir

Örnegin:

```
<%
Soldanbes = Left("ASP ögreniyoruz.", 5)
```

```
Response.Write soldanbes
%>
```

Sonucumuz "ASP ö" olacaktir.

Right(Degisken, Sayi): Bir yazi dizisindeki karakterlerin, sagdan "Sayi" kadarini gösterir. Kullanimi Left ile aynidir.

LCase(Degisken): Lower Case yani küçük harfli hal anlamindadir. Verdigimiz string'teki tüm harfleri küçük hallerine getirir.

```
<% Degisken = "ASP ögreniyoruz"
Response.Write Lcase(Degisken)
%>
```

Sonucu "asp ögreniyoruz" olarak görmekteyiz.

UCase(Degisken): Upper Case yani büyük harfli hal anlamindadir. Verdigimiz stringteki tüm harfleri büyük hallerine dönüstürür.

```
<% Degisken = "ASP ögreniyoruz"
```

```
Response.Write Ucase(Degisken)
%>
```

Sonucu "ASP ÖGRENIYORUZ" olarak görecegiz. Gerek Ucase, gerekse Lcase komutlari, sunucunun bölgesel ayarlarina göre degisebilir. VBScript temelinde ingilizce oldugundan, sorun çikarabilecek harf "i" ve "I" karakterleridir. "I" küçüldügünde "i" olacaktir. "i" harfi büyümeyebilir. Bu komutlarin saglikli islenmesi için bölgesel ayalari Türkçe olan sunucu seçmenizi tavsiye ederiz.

Trim, Ltrim, Rtrim: Bir degiskenin sagindaki, solundaki veya her iki tarafindaki bosluklari temizler. Formlarda yanlislikla basilan bosluklari yok etmek için kullanilir.

Örnegin degiskenimiz " ASP Ögreniyorum " olsun. Basinda 1, sonunda 2 bosluk olan bir dizi.

```
<%
Degisken = " ASP Ögreniyorum "
Response.Write Trim(Degisken)
%>
```

Sonucumuz "ASP Ögreniyorum" çikacaktir.

Response.Write Ltrim(Degisken) ile "ASP Ögreniyorum " , Rtrim(Degisken) ile " ASP Ögreniyorum" sonucunu alacagiz.

String(Sayi, degisken) : String komutu verilen degiskeni "Sayi" kadar çogaltir.

Örnegin:

```
<% Degisken = "Emre"
YeniString = String(3, Degisken)
Response.Write YeniString
%>
```

Alacagimiz sonuç "EmreEmreEmre" seklindedir.

VBScript'in Derinlikleri (Bölüm-5)

Mid: Bir degiskenin içinden baslangiç noktasini ve kaç karakter olacagini verdigimiz baska bir degiskeni verir.

Örnegin:

```
<% Degisken = "ASP ögreniyoruz"
YeniDegisken = Mid(Degisken, 4, 2)
Response.Write YeniDegisken
%>
```

"öğ" sonucunu alacagiz bu sorgudan sonra. Burada 4 baslangiç karakterimiz, 2 ise kaç karakter alacagimizi temsil ediyor.

Replace: Degiskende belirli karakterleri baskalariyla degistirmemize yariyor Replace komutu.

Örnek vermek gerekirse:

```
<% Degisken = "ASP ögreniyoruz"
YeniDegisken = Replace(Degisken, "uz", "m ben de")
Response.Write YeniDegisken
%>
```

Ekrana yazilacak sonuç "ASP ögreniyorum ben de" olacaktir.

Replace ile istediginiz karakterden itibaren kaç tane tanimlamaya uyan varsa onlari degistirebilirsiniz.

Örnegin

```
<% Degisken = "Kartal kalkar, dal sarkar"
YeniDegisken = Replace(Degisken, "a", "e", 6, 3)
Response.Write YeniDegisken
%>
```

Sonucunu "Kartal kelker, del sarkar" olarak alacagiz. 6. karakterden itibaren 3 tane "a" yi "e"ye dönüstürmeyi tanimladik. Replace komutu ayni zamanda belirli dizilerden istediklerimizi silmeye de yarayabilir.

```
<% Degisken = "Kartal kalkar, dal sarkar"
YeniDegisken = Replace(Degisken, "a", "")
Response.Write YeniDegisken
%>
```

Yeni Degiskenimiz "Krtl klkr, dl srkr" seklinde olacaktir.

VBScript'in Derinlikleri (Bölüm-6)

Test Fonksiyonlari

Sayfalar arasi veri akisinda, bazi degiskenler, geçirdikleri islemlerden sonra bir takim deformasyonlara ugrayabilirler. Örnegin, kullanici ile interaktif sayfalarda, kullanicilarin girecegi verileri belirlememiz bazen olanaksiz gibidir. String islemleri veya matematik islemler gibi bir dizi operasyondan geçirmemiz için, onlari istedigimiz forma getirmek zorundayiz. Tabi istedigimiz formda olup olmadiklarini sinamak istedigimizde, Test

fonksiyonlarini kullaniyoruz.

Bu fonksiyonlarda çikan iki tane sonuç vardir. True (1, Dogru), False(0, Yanlis)

Sonucumuz False çikarsa, verdigimiz tanim yanlis, True çikarsa verdigimiz tanim dogru olarak sinanacaktir.

IsArray: Degiskenlerin Array olup olmadigini test eder. Yani degisken birden fazla degiskeni daha içeriyor mu diye sinar.

IsDate: Degiskenin tarih olup olmadigini kontrol eder. Formlarda kullanicilarin tarih girmesi isteniyorsa bu fonksiyon kullanilarak, dogru veri girilip girilmedigi kontrol edilir.

IsEmpty: Degiskene ait degerin olup olmadigini sinar. Empty olmasi degiskenin degerinin "" 'e esit oldugunu belirler.

IsNull: Degiskenin degerinin geçerli olup olmadigini sorgular. Empry ile karistirilmamasi gereken bri tanimdir.

IsNumeric: Degiskenin sayi cinsinden olup olmadigini kontrol eder.

IsObject: Degiskende ActiveX veya OLE objeleriyle tanimlanip, tanimlanmadigini sorgular

TypeName: Degisken türünü bildirir.

VarType: Degisken türlerine ait sayiyi vermektedir.

Tür	Deger	Tanim
VbEmpty	0	**Empty** (Belirtilmemis veri
VbNull	1	**Null** (Geçerli olmayan veri)
VbInteger	2	Tam sayi
VbLong	3	Büyük tam sayi (Long integer)
VbSingle	4	Single kayan nokta sayisi
VbDouble	5	Double kayan nokta sayisi
VbCurrency	6	Para birimi
VbDate	7	Tarih
VbString	8	Alfanümerik satir (String)
VbObject	9	Obje

VbError	10	Hata
VbBoolean	11	Boolean (1 veya 0, bit, Yes/No)
VbVariant	12	Tür (Array türlerinde kullanilmaktadir)
VbDataObject	13	Veri erisim nenesi
VbByte	17	Byte
VbArray	8192	Array

VBScript'in Derinlikleri (Bölüm-7)

Bir örnek verelim.

```
<HTML>
<HEAD>
<TITLE>Fonksiyonlar</TITLE>
</HEAD>
<BODY BGCOLOR="#FFCC00">
<FORM METHOD=POST ACTION="kontrol.asp">
Adinizi Giriniz: <INPUT TYPE="TEXT" NAME="isim"
SIZE=25><BR>
```

```
Hangi yil dogdunuz <INPUT TYPE="TEXT" NAME="dogumyil"
SIZE=25><BR>
Hangi Ayda Dogdunuz <INPUT TYPE="TEXT" NAME="dogumay"
SIZE=25><BR>
Dogdugunuz gün: <INPUT TYPE="TEXT NAME="dogungun"
SIZE=25><BR>

<INPUT TYPE="Submit" NAME="Gonder" VALUE="Gönder">
</BODY>
</HTML>
```

Buradaki bilgilerin hepsi kontrol.asp'ye String olarak gönderilecektir.

Kontrol.ASP'yi yazarsak:

```
<%
Dim isim
Dim dogumyil
Dim dogumay
Dim dogumgun

Isim = Request.Form("isim")
```

```
Dogumgun = Request.Form("dogumgun")

Dogumay = Request.Form("dogumay")

Dogumyil = Request.Form("dogumyil")

If IsNumeric(Dogumay) = True Then

Response.Write "Dogum günü olarak rakam girdiniz"

Else

Response.Write "Dogum gününüz geçersiz."

End If

If IsNumeric(Dogumyil) = True AND Len(Dogumyil) = 4 Then

Response.Write "Dogum yilinizi dogru girdiniz."

Else

Response.Write "Dogum yiliniz geçersiz"

End If

If IsNumeric(Dogumay) = True AND Abs(Dogumay) <= 12 Then

Response.Write "Dogdugunuz Ayi rakamla yazmissiniz."

Else

Response.Write "Dogdugunuz Ayi harflerle yazmis olabilirsiniz

End If
```

Application ve Session Nesneleri

ASP kursumuza baslarken genel olarak ASP'ye neden gerek duyuldugundan bahsetmistik. CGI'in nerelerde yetersiz kaldigini, nerelerde servera gereksiz yük bindirdigini söylemistik.

Kullandiginiz Windows sistemi bir düsünün. Sifrenizi yazmanizla (veya Windows'un açilmasiyla beraber - Logon) size ait bir Oturum açilmaktadir (Session). Makinenizde çalistirdiginiz her program (Application) ile sisteminizi yönetirsiniz.

ASP de her web sitesini bir Application ve sitenize giren her kullaniciyi da birer Session olarak görmektedir. Söyle ki; kullanici web sitenize girdiginde bir session açilir. Yaptigi tüm hareketler bu Session içerisine dahildir ve siteden çiktiginda bu Session kapanir. Kisacasi kullanici web sitenizde sürekli takip edilir. Session kisiye özeldir. Yani Session ile tutulan bilgiler o kisi haricinde kullanilmaz.

Application nesnesi ile; site ile ilgili bilgiler tutulmaktadir. Sitenin uygulama yapisi, degiskenleri, nesne ve metodlari yine Application içerisindedir. Session'in tersi olarak bu bilgilere her kullanici

session'indan ulasilabilir.

Application ve Session'in genel özellikleri ilk basta global.asa dedigimiz açilis dosyasinda belirtilebilir.

Global.asa nedir?

Global.asa, sunucuya girildiginde veya sunucu açildiginda yapilacak isleri siralayan bir çesit ASP dosyasidir. Uzantisi ASP olmadigi gibi <% %> kodlari içerisinde de yazilmayan bir VBScript'tir diyebiliriz. Global.asa'yi yazarken dikkatli olmak gerekiyor. Keza yazilan hatali bir kullanim, tüm sitenizi bastan açip kapatmaniza neden olacaktir.

Global.asa'nin genel kullanimi

1. Web sitesi açildigi zaman,

2. Kullanici web sitesine girdigi zaman,

3. Kullanici web sitesinden çiktigi zaman yapilacak isleri düzenlemek için kullanilmaktadir.

Application ve Session Nesneleri (Bölüm-2)

Örnek bir GLOBAL.ASA

```
<SCRIPT LANGUAGE="VBScript" RUNAT="Server">
Sub Application_OnStart
Application.Lock
Application("Acilis") = Now()
Application.UnLock
End Sub

Sub Session_OnStart
Application.Lock
Application("sayac") = Application("sayac") + 1
Application.UnLock
End Sub

Sub Session_OnEnd
Application.Lock
Application("sayac") = Application("sayac") - 1
Application.UnLock
End Sub

</SCRIPT>
```

Yukarida görüldügü üzere, Session_OnStart, Application_OnStart ve Session_OnEnd rutinlerini kullandik ve Application("degisken") = deger seklinde Application nesnesine degerler atadik.

Application nesnesine girilen degisken degerleri, herhangi bir kullaniciya yolladiginiz web sayfalarinda ayni olarak gözükmektedir ve her kullanici bu bilgilere ulasabilmektedir.

Session'lar web sitesi ayarlarinda degistirilmedikçe, kullanici baglandiktan sonra 20 dakika site üzerinde herhangi bir islem, navigasyon yapmayinca kapanirlar.

Application ve Session Nesneleri (Bölüm-3)

Sitemizde iki kullanici bulunuyor olsun.

Global.asa'mizi sitemize koyduktan sonra ASP'mizi hazirlayalim

```
<% if Session("isim") = "" Then AND Request.Form("isim") = "" Then %>
<FORM METHOD=POST ACTION="<%= Request.ServerVariables("SCRIPT_NAME") %>">
<INPUT NAME="isim" TYPE="TEXT" SIZE=20>
```

```
<INPUT TYPE=Submit NAME="Onayla" VALUE="Onayla">
</FORM>
<% Else
if Request.Form("isim") <> "" Then
Session("isim") = Request.Form("isim")
End If

Response.Write "<B>Isminiz:</B> " & Session("isim")

Response.Write "<BR><BR>Su anda sunucuda " &
Application("sayac") & " kisi bulunuyor..."

End If
%>
```

Simdi 2 kullanici da isimlerini girsinler. Application objesinde sayaç tutuldugu için ikisinde de ayni gözükecektir. Fakat Session'lar kisiye özel oldugu için ikisi kullanici da yalnizca girdigi bilgileri görecektir.

Session'lar Cookie tabanli olsa da, kullanicilarin bilgisayarlarinda herhangi bir bilgi birakmazlar. Cookie özelligi sadece kullanicinin bagli olup olmadigini kontrol etmek için kullanilir. Yoksa her kullanicinin bilgileri, sunucuda ayri olarak tutulmaktadir.

HTML'dekinin aksine, sayfalar arasi bilgi aktarmak, formlar ve querystring'lerin yaninda Session'lar ile de mümkündür. Session'a bir degisken ve degeri atandiginda oturum kapanana kadar o deger kullanicinin gezdigi her sayfada kullanilabilir.

Örnegin:

DegerAta. ASP

```
<HTML>
<HEAD>
<TITLE>Session Degeri Atama</TITLE>
</HEAD>
<BODY BGCOLOR="#FFCC00">
<%
Session("deger") = Now()
%>
```

```
<a href="degeroku.asp">Deger Oku.asp</a>
</BODY>
</HTML>
```

degeroku.asp

```
<HTML>
<HEAD>
<TITLE>Session Degeri Okuma</TITLE>
</HEAD>
<BODY BGCOLOR="#FFCC00">
<H1>Degerata.ASP'ye girdiginiz Saat ve Tarih:</H1>
<% Response.Write Session("deger") %>
</BODY>
</HTML>
```

Degeroku.asp 'de farkedeceginiz üzere Degerata.asp'ye girdiginiz andaki tarih ve saati belirtmektedir.

FileSystem Nesnesi

FileSystem (Dosya Sistemi) nesnesi, ASP ile bulundugu veya

ulasabildigi sunuculardaki (LAN vb.) dosyalari yönetebilme özelligini sunuyor bizlere. Dosya sistemi nesnesi ile, sunucudaki dosyalari kopyalayabilir, düzenleyebilir, silebilir, yeni dosyalar ve klasörler yaratabilirsiniz.

Bir örnekle anlatmaya çalisirsak:

```
<HTML>
<HEAD>
<TITLE>Bir sürücü içeriginin okunmasi</TITLE>
</HEAD>
<BODY BGCOLOR="#FFCC00">

<% if Request.QueryString("dizin") = "" Then
anadizin = "C:"
Else
anadizin = Request.QueryString("dizin")
End If
%>
<font face="Verdana" size="2">
<H2>Dizin: <%= anadizin %></H2>
```

```
<%
if right(anadizin,1) = ":" Then
anadizin = anadizin & "\"
UstDizin = 0
Else
UstDizin = 1
End If

if right(anadizin,2) = ":\" Then
UstDizin= 0
End IF

Dim fso, d, s
Set fso = Server.CreateObject("Scripting.FileSystemObject")
Set d = fso.GetDrive(fso.GetDriveName("C:"))
s = "<b>Sürücü C: adi</b>: "
s = s & d.VolumeName & "<BR>"
s = s & "<b>Diskteki bos alan: </b>" &
FormatNumber(d.FreeSpace/1024, 0)
s = s & " Kbyte"
```

```
ShowFreeSpace = s

Response.write s


%>

<HR>

<% Set bf = fso.GetFolder(anadizin)

set pf = bf.ParentFolder

if UstDizin = 1 Then %>


<B><a href="<%= Request.ServerVariables("SCRIPT_NAME") &

"?dizin=" & pf %>">Üst Dizin: <%= pf %></a></b><br><br>


<% End If


Set fd = fso.GetFolder(anadizin)

Set fd2 = fd.SubFolders

For Each f1 in fd2

dizinler = dizinler & "<a href=" & chr(34) &

Request.ServerVariables("SCRIPT_NAME") & "?dizin=" & anadizin
```

```
& "\" & f1.name & chr(34) & ">" & f1.name & "</a>"
dizinler = dizinler & "<BR>"
Next

Response.Write dizinler
Set f = fso.GetFolder(anadizin)
Set fc = f.Files
For Each f1 in fc
dosyalar = dosyalar & f1.name
dosyalar = dosyalar & "<BR>"
Next
ShowFolderList = s
Response.Write dosyalar
%>
</BODY>
</HTML>
```

Bu örnegimizde dikkat edeceginiz üzere dosyalar arasi bir

navigasyon yarattik. Dizinlerin dolasimini saglayip, dizinler içindeki alt dizinleri ve dosyalari gördük.

FileSystem nesnesini kullanabilmek için öncelikle bu nesneyi yaratmamiz gerekiyor

```
<%

Dim FS

Set FS = CreateObject("Scripting.FilesSystemObject")

%>
```

FileSystem Nesnesi (Bölüm-2)

FileSystem objesi ile birlikte kullanabileceginiz komutlar	
CopyFile	Dosya kopyalar
MoveFile	Dosyalari baska bir dizine tasir
CopyFolder	Klasörleri kopyalar
MoveFolder	Klasörleri tasir veya ayni dizindeyse ismini degistirir

CreateFolder	Yeni klasör olusturur
DeleteFile	Dosya Sil
DeleteFolder	Klasörleri siler
FileExists	Dosyanin olup olmadigini sinar
FolderExists	Klasörün olup olmadigini sinar
DriveExists	Sürücünün olup olmadigini sinar
CreateTextFile	Yeni bir metin belgesi olusturur.
GetFolder	Klasörü ve içerigini kullanima hazirlar
GetFile	Dosyayi kullanima hazirlar
GetDrive	Sürücüyü kullanima hazirlar
GetFolderName	Dizin ismini verir
GetDriveName	Sürücü ismini verir
GetParentFolderName	Üst dizinin adini verir
GetFileName	Dosya adini verir.

FileSystem nesnesinde Wildcard (*) isaretleri kullanilabilir örnegin C: sürücünüzde test diye bir dizin açin. Bu dizine yedegini aldiginiz resimler, metin belgeleri, çalistirilabilir dosyalar atin.

```
<%

Dim FS
```

```
Set FS = CreateObject("Scripting.FilesSystemObject")
FS.DeleteFile "c:\test\*.jpg"
%>
```

Bu kod, C: sürücüsünün test dizini içerisindeki tüm JPG uzantili dosyalari silecektir.

```
<%
Dim FS
Set FS = CreateObject("Scripting.FileSystemObject")
FS.CreateFolder "c:\CHIPbackup"
%>
```

Bu kod ile de farkedeceginiz üzere C: sürücüsünde CHIPBACKUP seklinde bir klasör açmis oluyoruz. Simdi daha önceki test klasörümüzdeki dosyalari yeni açtigimiz "CHIPbackup" klasörümüzde kopyalayalim.

```
<%
Dim FS
Set FS= Createobject("Scripting.FileSystemObject")
```

```
FS.CopyFile "c:\test\*.*", "c:\CHIPbackup\"
%>
```

FileSystem nesnesinde bir tek özellik bulunmaktadir. Drives özelligi bir çok dizi ve degisken kabul edebilecegimiz kolleksiyonlardan olusmaktadir. Bu kolleksiyonlari verdigimiz ilk örnekte görebiliyorsunuz. Ama simdi detayli olarak inceleyelim.

Kolleksiyonumuz: FileSystem.Drives

Drive isimleri: Kolleksiyon.VolumeName

Drive Harfleri: Kolleksiyon.DriveLetter

Bos Alan (Byte cinsinden): Kolleksiyon.FreeSpace

Sürücü Tipi: Kolleksiyon.DriveType

Örnegin:

```
<% Dim FS, Drv, Drvs Set FS =
Server.CreateObject("Scripting.FileSystemObject") Set
Drvs = FS.Drives For each Drv in Drvs %>
```

Sürücü Tipleri

Sonuç	Açiklama
0	Bilinmiyor
1	Çikarilabilir (Removable)

2	Sabit
3	Ag sürücüsü
4	CD-ROM
5	RAM-Drive

<B>Sürücüler: </B> <%= Drv.DriveLetter %>

<% if Drv.IsReady = True Then %>

<B>Sürücü Adi: </B> <%= Drv.VolumeName %>

<B>Sürücüdeki Bos Alan: </B> <%= Drv.FreeSpace %>

<% Else %>

<B>Sürücü Hazir Degil</B>

<% End If

Next %> Diger kolleksiyonlar ise Folders (Dizin), SubFolders(Alt-Dizin) ve Files(Dosyalar) kolleksiyonlaridir.

Bu kolleksiyonlardaki özelliklere gelince:

Sürücü Tipleri	
Özellik	Sonuç
Name	Dizin veya dosyanin adini vermektedir

DateCreated	Yaratildigi tarihi verir
DateLastAccessed	Son erisim tarihi
DateLastModified	Son degistirilme tarihi
Size	Dizin ise alt-klasörler ve dosyalarla birlikte boyutu
Drive	Bulundugu sürücü
ParentFolder	Bulundugu dizinin adi
SubFolders	Alt-dizinleri
IsRoot	Kök dizin olup olmadigi

Simdi ilk basta verdigimiz örnegimize tekrar dönersek, orada GetFolders metoduyla özellikleri aldigimizi ve Name özelligiyle dosya ve klasör isimlerini ekrana yazdirdigimizi göreceksiniz.