Ders1: JS Nedir?

Javascript, Netscape firması tarafından oldukça çok duyulan C dilinden türetilmiştir.Javascript, html gibi, komutları düz yazı dosyası olarak kaydedilen ve işleyebilmesi için bir yorumlayıcıya (browsera) ihtiyacı olan bir dildir.Javascript, dinamik sayfalar yapma ihtiyacı ortaya çıkmaya başladığı zaman üretilmiş bir programlama dilidir. Javascript'te ait özelliklri maddeler halinde özetlersek;

* Javascript, html in içine gömülür, burada browserlar tarafından yorumlanır.
* Javascript browsera bağımlı olarak yorumlanır, bazı javascript komutları farklı browserlarda farklı şekilde yorumlanabileceği gibi, bazen hiç bir etki yaratmaz.
* Html içindeki javascript kodlarının browserlar tarafından algılanması için, kodlar <script>.......</script> etiketleri içine yazılır.Ya da ayrı bir dosyada yazılarak ".js" uzantısıyla kaydedildikten sonra, gerekli duyulduğu yerde bağlantı yolu ile çağırılır.
* Javascript, ancak bir olay halinde işler.Burada olaydan kasıt, sayfanın açılması, kullanıcının klavyeden değer girmesi, bir butona tıklaması ve benzeri olaylar olabilir.

Web Browser programları, Javascript komutlarını yorumlayabilmek için, html in içinde <SCRIPT LANGUAGE="Javascript"> etiketini ararlar ve </SCRIPT> etiketini gördükleri anda bu yoruma son verip, html yorumuna dönerler.Bu etikette, kullanılan javascript sürümü de belirtilir.Netscape ileri sürümlere ait komutları tanıyorsa icra eder, tanımıyorsa görmezden gelir. IE ise belirtilen sürümden ileri komutları tanımaz ve hata mesajı verir.Şu an en güvenli sürüm 1.2’dir.Örneklerde de bu sürüm belirtilecektir.Javascript’in birinci sürümünü kastetmek amacıyla <SCRIPT LANGUAGE="Javascript1.0"> veya "<SCRIPT LANGUAGE="Javascript1"> yazılmaz, sadece "Javascript" yazılır.

Javascript kodlarının başladığını ve sürümünü belirtmek için, <SCRIPT LANGUAGE="Javascript1.2"> kullanıldığı gibi son zamanlar ikinci bir kullanım şekilde vardır.Bu kullanım şekli ise şöyledir; <SCRIPT type="text/javascript" language="JavaScript1.2">  
Sizde yapacak olduğunuz uygulamalarda istediğiniz bir tanesini kullanabilirsiniz.

Ders2: JS’e giriş

Bu dersimizde yapacağımız basit örnekler ile javascript in temel kullanımını anlamaya çalışalım.

Örnekte **<**script language="Javascript1.2">...</script> arasında **document.write("...")** diye bir tane javascript kodu bulunmaktadır.Bu kod, parantez içine yazılan html kodlarını uygular.Aslında ilk "Merhaba" yazısında bunu kullanmaya gerek yoktur.Gerekli olan kullanım şekli ikincisidir, ama; değişik kullanım şekillerini görmeniz için her ikisi de verilmiştir. "Merhaba<br>" deki <br> etiketi bir alt satıra inmek için kullanılmıştır.

|  |
| --- |
| <html>  <head>  <title>Javascript te Giris</title>  </head>  <body>  <script language="Javascript1.2">  document.write("Merhaba<br>")  document.write("<font size=6 color=blue>Merhaba</font>")  </script>  </body>  </html> |

Javascript te küçük bir hata, sayfanın çıkmamasına veya pencerenin alt kısmında javascript hatası belirmesine neden olur.Javascript, html gibi esnek bir dil değildir.Bu nedenle yapacağınız örneklerde yazım hatalarına dikkat edin.

Javascript yukarıdaki uygulamada olduğu gibi <body>...</body> etiketleri arasında yer alabileceği gibi, <head>...</head> arasında da yer alabilir.Bunu fonksiyonlu bir örnekle göstermeye çalışalım.

|  |
| --- |
| <html>  <head>  <title>Javascript Uygulamasi</title>  <script language="Javascript1.2">  function merhaba()  {  alert("Merhaba, Javascript")  }  </script>  </head>  <body onLoad="merhaba()">  </body>  </html> |

Programlama bilen kişilerin javascripti anlaması daha kolaydır.Bu örnekte programlama dillerinin vazgeçilmezi olan fonksiyon kullanılmıştır.Örnekte javascript kodu <head> bölümünde fonksiyon halinde yer almaktadır. **function merhaba()** ifadesi ile merhaba() isimli bir fonksiyon tanımlanmış, bu fonksiyonun ne yaptığı süslü parantezler arasına yazılmıştır. Fonksiyon içindeki**alert("...")** kodu, javascript te kullanılan bir mesaj kutusudur.<body> etiketinde **onload="..."**durumu atanmış.Bu "yüklendiği takdirde"anlamına gelir.Yani sayfa yüklenildiğinde merhaba() mesajı verilmektedir.Tüm bunlara ilerleyen derslerde daha detaylı şekilde değinilecektir.

Javaccript kodu, css de olduğu gibi, dış bir dosyayla da html e tanıtılabilinir.Javascript dosyası **.js**uzantılıdır. Aşağıda merhaba.js isimli dosyanın içeriği verilmiştir.

                                            document.write("<center>")  
                                            document.write("<h1>Merhabalar!</h1>")  
                                            document.write("</center>")

Şimdi bu javascript dosyasının sayfa içinde nasıl çağırıldığına bakalım.

|  |
| --- |
| <html>  <head>  <title>Javascript Uygulamasii</title>  <script src="merhaba.js">  </script>  </head>  <body>  </body>  </html> |

Ders 3: JS Yazım Kuralları

Browserların javascript yorumlama bölümleri, kodların arasında yer alan yorum ifadelerini yani açıklamaları ve boşlukları atarlar.Geri kalan kelimeleri beşe ayırarak ele alırlar.

**1. Belirleyiciler :**

Javascript dilinin değişkenleri, metodları, fonksiyonları ve nesnelerini belirleyen isimlere belirleyicilerdenir.Belirleyiciler ya harfle ya da alt çizgi (\_) ile başlar.Rakam veya diğer işaretler ile başlayamazlar.Javascript, küçük, büyük harf ayrımı yapan bir dildir.Belirleyicilerin içinde boşluk olamaz.Bu kuralların yanısıra, belirleyici isimlerinde türkçe karakter kullanmamaya dikkat edilmelidir.

Birkaç belirleyici kelime örneği:

|  |  |
| --- | --- |
| Doğru | Yanlış |
| \_sayi | #sayi |
| ikincisayi | 2incisayi |
| arasinav | ara sinav |

Javascript in küçük, büyük harf ayrımı yapan bir dil olduğunu az önce söylemiştik.Bu nedenle arasinav ile Arasinav farklı belirleyiciler olarak ele alınır.Ayrıca anahtar kelimeler ve ayrılmış kelimeler de belirleyici ismi olarak kullanılamaz.

**2. Anahtar Kelimeler :**

Javascript dilinin önceden tanımlanmış ve programın yorumunda özel anlam kazandırılmış kelimelerine anahtar kelime denilir.Javascript e ait anahtar kelimelr aşağıda verilmiştir.

|  |  |  |
| --- | --- | --- |
| break | continue | else |
| false | for | function |
| if | in | int |
| new | null | return |
| this | true | var |
| while | with | typeof |
| void | do | labeled |
| switch |  |  |

**3. Ayrılmış Kelimeler :**

Javascrit te bazı kelimeler ayrılmış kelimeler olarak adlandırılır ve bu kelimelerin belirleyici ismiolarak kullanılmasına izin verilmez.Bu kelimeler javascript tarafından rezerv edilmiş kelimelerdir.Javascript te kullanmayacağınız kelimelerin listesi:

|  |  |  |
| --- | --- | --- |
| abstract alert arguments Array blur boolean Boolean break byte callee caller captureEvents case catch char class clearInterval clearTimeout close closed confirm  const constructor continue Date debugger default defaultStatus delete do document double else enum escape eval export extends false final finally  find float focus for frames | Function function goto history home if implements import in infinity innerHeight innerWidth instanceof int interface isFinite isNaN java length location  locationbar long Math menubar moveBy moveTo name NaN native netscape new null Number Object open opener outerHeight outerWidth package Packages pageXOffset  pageYOffset parent parseFloat parseInt personalbar | print private prompt protected prototype public RegExp releaseEvents resizeBy resizeTo return routeEvent scroll scrollbars scrollBy ScrollTo self setInterval setTimeout short static status statusbar stop String super switch synchronized this throw throws toolbar top toString transient true try typeof unescape unwatch valueOf var void watch while window with |

**4. Değerler :**

Javascript kodu icra edildiği sırada değişmeyen rakam veya metinlere değer denir. Javascript kodlarında beş tür değer bulunur:

4-a. Tamsayı Değerler :

Tamsayılar, 10 tabanlı, 8 tabanlı (octal) veya 16 tabanlı (hexadecimal) olabilir.8 tabanlı sayıları belli etmek için sayıdan önce sıfır, 16 tabanlı sayıları belli etmek için "0x" kullanılır. Örnekler : Decimal: 12594      Octal: 0727898      Hexadecimal: 0x68ac

4-b. Kesirli Değerler :

Kesirli değerler, tamsayı ve kesirli kısım olmak üzere iki parçadan meydana gelir.Kesir kısmını belirtmek için arada nokta kullanılır.Örnek : 125.45

4-c. Boolean Mantık İfadeleri :

Javascript te, iki boolean değeri vardır.Bunlar: True (Doğru) ve False (Yanlış)tır.Javascript, True (Doğru) değerini 1, False (Yanlış) değerini 0 rakamıyla tutar.Boolean Mantık İfadeleri, küçük harflerle yazılarak kullanılmalıdır.

4-d. Alfanümerik (Karakter) Değerler :

Çift-tırnak (") veya tek-tırnak (‘) içine alınan her türlü ifade, javascript için karakter değeridir.  
Örnekler : "Merhaba"  ‘Merhabal’  "8 tane"  ’125698’ gibi.

4-e. Özel Karakterler :

Javascript te bazı özel karakterleri kullanmak istediğinizde, bunu javascriptin anlaması için karakterlerin önüne ters bölü(\) işareti koymanız gerekir.Bu tür özel karakterler şunlardır:

|  |
| --- |
| \b - Klavyede Geri (backspace) tuşunun görevini yaptırır. |
| \f - Yazıcıya sayfayı bitirmeden çıkarttırır. |
| \n - Yazı imlecini yeni bir satırın başına getirir (new line). |
| \r - Klavyede Enter-Return tuşunun görevini yaptırır. |
| \t - Sekme (tab) işaretini koydurur. |
| \\ - Yazıya ters-bölü işareti koydurur. |
| \’ - Yazıya tek-tırnak işareti koydurur. |
| \" - Yazıya çift-tırnak işareti koydurur. |

Javascript’e bu tür özel karakterlerle html sayfasına bir metin yazdıracağınız zaman, bu yazının <PRE>..</PRE> etiketleri arasında olması gerekir. Aksi taktirde Javascript ne yazdırırsa yazdırsın, html bu özel karakterleri dikkate almaz.

Örnek :

<pre>  
<script language="Javascript1.2>  
document.writeln("Karakter değerleri çift tırnak(\") içinde yazılır.")  
</script>  
</pre>

**Javascript'te Açıklama Koyma :**

Javascript ile yapacağınız programlar bazen büyük ve karmaşık olurbilir.Gerek sonradan hatırlamak için, gerekse bizden başkalarının kodları okuyarak anlamalarına yardımcı olmak için, kodların arasına bazı notlar yazılabilir.Bu tür notları doğrudan yazılırsa, javascript hata verir veya programın işlevi değişir. Bu notları yazmak için not satırı önüne **//** işareti konur. Bu işaret bir satırlık not bırakmamızı sağlar ve javaccript kodlarını etkilemez.Bazen yazılack not birkaç satırdan oluşabilir.Bu durumda not **/\*...\*/** işaretleri arasına yazılır.

Ders 4: Değişkenler

Değişken, içinde değerler saklayabileceğiniz bir programlama aracıdır.Değişkenlerin bir adı bir de değeri olur.Program boyunca değişkenin adı değişmez, fakat içeriği değişebilir.

Değişkenlere isim verirken belirleyici isimleri kurallarına uymak gerekir.Yani bir değişkenin adı karakter veya alt çizgi dışında bir işaretle başlayamaz. Javascript, büyük harf-küçük harf ayırt ettiği için SAYI ve sayi kelimeleri iki ayrı değişken olarak algılanır.Bir değişkeni nasıl tanımladıysanız, program boyunca hep aynı şekilde yazmanız gerekir.

Değişken tanımlamak, bilgisayar programcılarına daima gereksiz bir yük gibi görünür. Birazdan göreceğiz, Javascript sadece değişkenleri tanımlamayı zorunlu kılmakla kalmaz, fakat nerede tanımlandığına da özel bir önem verir.

Javascript te değişken tanımlamak için **var** anahtar kelimesi kullanılır.

Örnekler:

var sayi  
var vize, final  
var siraNo, okulNo, sinif

Görüldüğü gibi aynı satırda araya virgül koyularak birden çok değişken tanımlanabilir. Javascript te değişken tanımlama esnasında değer atamakta mümkündür.

Örnekler:

var sayi = 10  
var cevap = true  
var siraNo = 3, okulNo = 1452  
var ders = "Web Tasarımı"

Javascript, programcıya bir değişkeni tanımlamadan kullanma imkanı da verir.Bu durumda o değişken genel (global) nitelik kazanır; ama iyi bir programlama ve özellikle daha sonra hata arama zorunluğu sebebiyle, değişkenlerin açık seçik tanımlanması en doğru yoldur.

Javascript'te beş tür değişken bulunur.

**1. Sayı Değişkeni:**

Sayı türündeki değişkenler, tam sayı, ondalık sayı, pozitif sayı veya negatif sayı olabilir.

Örnekler:

var sayi1 = 5, sayi2 = 12  
var katsayi = 0.707, baslangic = -12

**2. Boolean Değişken:**

Javascript, tanımlanırken değeri true veya false olarak belirtilen değişkenleri otomatik olarak boolean değişken olarak sınıflandırır.

**3. Alfanümerik (karakter) Değişken:**

Alfanümerik değişken, kendisine tek veya çift tırnak içinde verilen bütün karakterleri tutar.

Örnekler:

var adi = "Mehmet", Bolum = "Bilgisayar"  
var mesaj = "Merhabalar! Kolay gelsin."

**4. İşlev (Function) Değişken:**

Javascript’e genellikle fonksiyonlarla iş yaptırılır.Kimi fonksiyonları, javascript’i tasarlayanlar tanımlamıştır; bunlara metod denir.(Örnek: alert metodu).Kimi fonksiyonları biz tanımlarız ve komutlarını biz veririz.Bunlara ise programlanan fonksiyonlar denir.

**5. Nesne (Object) Değişkenleri:**

Bu tür değişkenlere değişken adı vermek bile gerekmez; çünkü bunlar browserın nesneleridir. Fakat javascript kodlarımızda bu nesneleri de değişken gibi kullanabiliriz.Document, window bunlardan ikisidir.Bu sınıfa giren özel bir değişken ise değerini "null" (içi boş) kelimesiyle belirlediğiniz değişkenlerdir.

Dersimizin başında javascriptin değişkenleri nerede tanımladığımıza dikkat ettiğini söylemiştik.Şimdi bunu bir örnekle açıklayalım.

|  |
| --- |
| <html>  <head>  <title>Degisken Durumlari</title>  <script language="Javascript1.2">  var metin1 = "Merhaba!!!"  function merhaba()  {  alert(metin1)  }  function selam()  {  var metin1 = "Sana da merhaba!"  alert(metin1)  }  </script>  </head>  <body>  <form>  <input type="button" value="Merhaba" onClick="merhaba()"><br>  <input type="button" value="Selam ver" onClick="selam()">  </form>  </body>  </html> |

Örneğimizde metin1 isimli değişken başta tanımlanmıştır.Bu değişken merhaba() fonksiyonunda kullanıldığında başlangıçtaki değeri dikkate alınır.selam() fonksiyonunda ise, ismi metin1 olan yeni bie değişken tanımlanmıştır.Buradaki değişken özel bir değişkendir. yani sadece tanımlandığı selam() fonksiyonu içinde anlamlıdır.Diğer tüm yerlerde en başta tanımlanan ve değeri "Merhabalar!!!" olan metin1 değişkenidir.

Ders 5: Diziler

Diziler programcılığın temel konularından biridir.Javascript in şu ana kadar ele aldığımız bütün değişkenleri tekildir.Yani bu değişkenlerden javascript bakımından sadece bir adet vardır. Dolayısıyla bir adet de değeri bulunur.

**Dizi Tanımlama**

Dizileri değişkenler kümesi şeklinde tanımlayabiliriz.Bellekte aynı isim altında saklanan değerleri ifade eder;

değerler **new Array(...)** kodu kullanılarak girilir.Dizilerin iki yazım türü vardır:  
         **var** dizi\_adı **= new Array(**değer1,değer2,değer3,değer4,...,değerN**)**

         veya

**var**dizi\_adı **= new Array(**N**)**

dizi\_adı[0] = değer1

dizi\_adı[1] = değer2

dizi\_adı[2] = değer3

dizi\_adı[3] = değer4

         ...........

          dizi\_adı[N] = değerN

Diziler sıfırdan itibaren numaralanır.Bu nedenle yukarıdaki dizi tanımlamalarında N+1 adet değer vardır.

|  |
| --- |
| <html>  <head>  <title>Diziler</title>  </head>  <body>  <script language="Javascript1.2">  var gunler = new Array("pazartesi", "sali", "çarsamba")  document.write(gunler[0] + "br ")  document.write(gunler[1] + "br ")  document.write(gunler[2] + "br ")  </script>  </body>  </html> |

Dizilerin taşıdığı değer sayısını ifade etmek için **length** komutu kullanılır.Diziyi yazılım içinde değer vermişsek, o dizinin kaç değeri olduğunu biliriz ve değer sayısını bulmaya çalışmayız. Ama eğer diziye girilecek değer sayısını kullanıcıya bırakmışsak, dizinin taşıdığı değer sayısını yazılım içinde ifade etmek isteriz.Örnekte <script> etiketleri arasına document.write(gunler.length) komudunu ekleyerek, dizinin kaç değere sahip olduğunu görebilirsiniz.

Ders 6: JS’te İşlemler

Program yazmanın amacı, bilgisayara işlem yaptırmaktır.Javascript'te bir programlama dili olduğuna göre, işlemler sıkça kullanılır.

**Atama İşlemleri**

Javascript’te en sık yapılan işlem, bir değişkene değer atama işlemidir.Javascript, bir eşittir işaretiyle karşılaştığında, önce işaretin sağına bakar ve değeri okur, sonra işaretin soluna bakar ve bu buradaki değişkenin tanımlı olup olmadığına bakar.Değişken tanımlı ise sağdaki değer, soldaki değişkene aktarılır.

Örnek:

var a = 5, b=10 , sonuc  
sonuc = a\*b

**Aritmetik İşlemler**

Aritmetik işlemlerde çarpma, bölme ve kalanlı bölme işlemlerinin toplama ve çıkarma işlemine göre önceliği vardır.Bir işlemde eğer hiç parantez yoksa, soldan sağa doğru sırasıyla çarpma ve bölme işlemleri yapılır.

Javascript'te yer alan aritmetik işaretler ve örnekleri:

|  |  |
| --- | --- |
| **+** | Toplama işlemini yapar. |
|  | sayi = 10 + 5 (sayi=15 olur) |
| **-** | Çıkartma işlemini yapar. |
|  | sayi= 10 - 5 (sayi=5 olur) |
| **\*** | Çarpma işlemini yapar. |
|  | sayi = 10 \* 3 (sayi=30 olur) |
| **/** | Bölme işlemi yapar. |
|  | sayi= 10 / 3 (sayi=3.3333333333333335 olur) |
| **%** | Tamsayılı bölme işlemi yapar ve kalanı verir. |
|  | sayi=10 % 3 (sayi=1 olur) |

 Aritmetik işlemlerde kısaltmalar kullanılabilir.

x = x + y  işlemini kısaltmak için x += y  
x = x - y   işlemini kısaltmak için x -= y  
x = x \* y  işlemini kısaltmak için x \*= y  
x = x / y   işlemini kısaltmak için x /= y  
x = x % y işlemini kısaltmak için x %= y kullanılabilir.

|  |
| --- |
| <html>  <head>  <title>Islemler</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var vize = "55", final,ort  final = 75  ort = (vize \* 0.4 + final \* 0.6)  document.write("vize = " + vize +"\n")  document.write("final = " + final +"\n")  document.write("Ortalama = " + ort)  </script>  </pre>  </body>  </html> |

Aritmetik işlemlerde sıkça kullanılan diğer kısaltmalar ise şöyledir.

x++  veya  ++x  -> x = x + 1 işlemi ile aynı anlama gelir.  
x--    veya   --x   -> x = x - 1 işlemi ile aynı anlama gelir.

Ders 7: Karşılaştırma İşlemleri ve Alfanümerik İşlemler

**Karşılaştırma İşlemleri**

Javascript'te yazılan programlarda bir yerlerde karar vermek gerekebilir.Karşılaştırma işlemleri bu amaçla kullanılır.Bu tür komutlarda javascript, sözünü ettiğiniz değişkenin değerini bulup, onu verdiğiniz bir başka ölçütle karşılaştıracak ve varacağı sonuca göre emrettiğiniz işi yapacaktır.

Karşılaştırma operatörleri ve işlevleri:

|  |  |
| --- | --- |
| **==** | Eşit operatörü. |
| !**=** | Eşit değil operatörü. |
| **>** | Büyük operatörü. |
| **<** | Küçük operatörü. |
| **>=** | Büyük veya eşit operatörü. |
| **<=** | Küçük veya eşit operatörü. |

Javascript’in karşılaştırma operatörleri genellikle "if" (eğer..ise) ifadesiyle birlikte kullanılır.Bununla ilgili detaylı açıklamayı daha sonraki derslerimizde vereceğiz; ama şimdi karşılaştırma işlemlerinin nasıl yapıldığını anlamak için basit bir örnek verelim.

|  |
| --- |
| <html>  <head>  <title>Karsilastirma Islemleri</title>  </head>  <body>  <script language="Javascript1.2">  var not=65  if(not >= 60)  {document.write("Dersi geçtiniz!")}  else  {document.write("Dersten Kaldiniz!")}  </script>  </body>  </html> |

Örneğimizde, not ismli bir değişken tanımlanmış ve değişkene 65 değeri atanmıştır.Karşılaştırma işleminde not değişkenine ait değerin 60'tan büyük veya eşit olması durumunda "Dersi geçtiniz!", diğer durmlarda ise "Dersten Kaldınız!" yazısının yazdırlması istenmiştir.Sizde örnekteki not değerini veya karşılaştırma işlemini değiştirerek konuyu daha iyi anlayabilirsiniz.

**Alfanümerik İşlemler**

Javascript alfanümerik değişkenlerin değerleri ile sadece toplama işlemi yapabilir.Aslında bu işlem toplama değil birleştirme, ekleme işlemidir.Şimdi bunu bir örnek ile gösterelim.

|  |
| --- |
| <html>  <head>  <title>Alfanümerik Islemler</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var m1 = "Pamukkale"  var m2 = "Üniversitesi"  var m3 = "Teknik Egitim Fakültesi"  document.write(m1 + m2 + "\n")  document.write(m1 + " " + m2 + "\n")  document.write(m1+ " "+ m2 + "-" + m3 + "\n")  </script>  </pre>  </body>  </html> |

Örnekte görüldüğü gibi alfanümerik değere sahip değişkenler toplama işareti ile birbirine eklenmektedir.

Alfanümerik değerli değişkenlerde karşılaştırma işlemlerinde kullanılabilir.

|  |
| --- |
| <html>  <head>  <title>Alfanümerik-Karsilastirma Islemleri</title>  </head>  <body>  <script language="Javascript1.2">  var m1 = "PAU"  if(m1=="PAU")  {alert("Pamukkale Üniversitesi-PAU")}  else  {alert("Baska bir üniversite")}  </script>  </body>  </html> |

Ders 8: Şartlı İşlemler ve Mantıksal İşlemler

Javascript’te karşılaştırma yaparken şartlı işlemler de yaptırabilirsiniz.Şartlı işlemler (..ise ..yap!) mantığına göre çalışır.Şartlı işlemlerde ? (soru işareti) ve : (iki nokta üstüste) işaretleri kullanılır.

değişken = (şart)**?**"Şartın sağlandığı durum " **:** "Şartın sağlanmadığı durum "

|  |
| --- |
| <html>  <head>  <title>Sartli Islemler</title>  </head>  <body>  <script language="Javascript1.2">  var x = 20, y = 25  var sonucmsj  sonucmsj = (y==x)?"y degiskeni x degiskenine esit!" : "y degiskeni x degiskenine esit degil!"  alert(sonucmsj)  </script>  </body>  </html> |

x değeri ile y değeri eşit olmadığı için şart sağlanmaz.Bu nedenle sonucmsj değişkenine, "y degiskeni x degiskenine esit degil!" değeri atanır.

**Mantıksal İşlemler**

Mantıksal işlemler karşılaştırmalarda sıklıkla kullanılır.Bu sayede aynı yapıda, birden çok şarta bakılabilir.Javascript'te kullanılan mantıksal sorgulama işaretleri şöyledir.

|  |  |
| --- | --- |
| **&&** | Mantıksal Ve: iki koşul da doğru olmalı. |
| **||** | Mantıksal Veya: ya birinci, ya da ikinci koşul doğru olmalı. |
| **!** | Mantıksal Değil: Tek koşula uygulanır.Değilse anlamındadır. |

|  |
| --- |
| <html>  <head>  <title>Mantiksal Islemler</title>  </head>  <body>  <script language="Javascript1.2">  var para = 65  var kredi = "yok"  if((para >= 60) || (kredi = "var"))  {document.write("Ürünü satin alabilirsiniz.")}  else  {document.write("Üzgünüm! Paraniz yetmiyor yada krediniz yok.")}  </script>  </body>  </html> |

if yapısında veya işareti kullanıldığı için şartlardan bir tanesinin sağlanması yeterlidir.Örneğimizde para değeri 60'tan büyük olduğu için kredi olmadığı halde şart sağlandı.Bu nedenle şartın sağlanması durumunda görüntülenmesi istenilen yazı yazdırıldı.

Ders 9: Akiş Denetimi

Javascript veya bir başka programlama dili ile yazacağınız programda, zaman zaman programın akışını, programa bırakmanız gerekir.Programcı olarak siz, programın izleyeceği alternatif yolları belirlersiniz; fakat bu yollardan hangisini izleyeceğine, sizin koyduğunuz şartları değerlendirerek program kendisi karar verir.

Javascript'te akış denetimi için kullanılan iki deyim vardır:İlki **if** deyimi, ikincisi **switch** deyimidir.

**"if... else..." Deyimi**

Program akışını denetlemekte en çok kullanılan deyim if deyimidir.if deyimi tek başına kullanılabileceği gibi if.... else... şeklinde de kullanılabilir.if deyimi tek başına kullanıldığında, if deyiminden sonraki şarta bakılır.Eğer şart sağlanıyorsa, tanımlanan işlemler yapılır; sağlanmıyorsa, işlemler atlanır.if.... else... yapısında ise, if'teki şart sağlanmıyorsa, else kısmında tanımlanan işlemler yapılır.

Genel Kullanımı:

**if (**koşul ifadesi**)** işlem

**else** işlem

birden fazla işlem söz konusuysa,

**if (**koşul ifadesi**)**

**{**

İŞLEMLER

**}**

**else**

**{**

İŞLEMLER

**}**

Şimdi form araçları kullanarak if...else.... yapısının nasıl çalıştığını anlamaya çalışalım.Örnekteki fonksiyon kullanımı ve olaylar konusu daha sonraki derslerimizde anlatılacaktır.Siz if...else.. yapısının yaptığı işe dikkat edin.

Örneğimiz kullanıcıdan bir şifre girmesini istiyor.Daha sonra girilen şifre değeri if deyimindeki şart ile karşılaştırılıyor.Eğer girilen şifre değeri "pau" ise ekrana "Pamukkale Üniversitesine Hosgeldiniz!" mesajı, diğer durumlarda ise "Lütfen dogru sifreyi giriniz!" mesajı veriliyor.

|  |
| --- |
| <html>  <head>  <title>if Deyimi</title>  <script language="Javascript1.2">  function kontrol(form)  {  var sifre = form.sifre.value  if(sifre == "pau")  {alert("Pamukkale Üniversitesine Hosgeldiniz!")}  else  {alert("Lütfen dogru sifreyi giriniz!")}  }  </script>  </head>  <body>  <form>  <input type="text" name = "sifre">  <input type="button" value ="Tamam" onClick="kontrol(this.form)">  </form>  </body>  </html> |

Burada **alert** komutundan kısaca bahsedelim.alert komutu uyarı mesajı vermeye yarayan bir komuttur. Verilmek istenilen mesaj, parantezler içine yazılır.Genel kullanımı şöyledir.

**alert(**Gösterilecek mesaj**)**

Eğer gösterilecek mesaj alfanümerik(karakter) ise, çift tırnak içinde yazılır.

**"switch" Deyimi**

**if...else...** koşul deyiminde yol ayrımı söz konusudur ve akış ikiye ayrılır. Şimdi göreceğimiz **switch**kodu ise akışı ikiden fazla sayıya ayırır. Yani birden fazla koşul sözkonusudur.

Genel Kullanımı:

**switch (**ifade)

**{**

**case** değer1**:** işlem1

**break**

**case** değer2**:** işlem2

**break**

**...**

**case** değerN**:** işlemN

**break**

**default:** işlem

**}**

birden fazla işlem söz konusuysa,

**switch (**ifade)

**{**

**case** değer1 **:**

**{**

işlemler

**}**

**break**

**case** değer2 **:**

**{**

işlemler

**}**

**break**

**...**

**case** değerN **:**

**{**

işlemler

**}**

**break**

**default:**

**{**

işlemler

**}**

**}**

Bu koşul eyleminde değişken **switch(...)** kısmına yazılır. Sonra değişkenin alabileceği değerler **case:** (durum) kodlarının yanlarına yazılarak koşul ifadesi yaratılır ve alttaki işlemler uygulanır. Eğer case: kodlarından hiçbiri değişkenin taşıdığı değeri tutmuyorsa, **default:** kodunun altındaki işlemler uygulanır.(default kodunun yazılması zorunlu değildir)Buradaki **break** (kes) komutu önemlidir.İşlemlerin altına break kodu yazmazsak, bir alttaki case (durum) koşuluna ait işlemler de yapılır. break kodunu işlemlerin altına koyarsak, işlemler bitince bir alttaki case koşulunun işlemlerine geçilmez ve program switch deyiminin sonundan devam eder. Buradan hareketle aynı işlemlere tabi tutulacak koşul ifadelerini alt alta yazın ve bu ifadelerden en altında olanın işlemlerine break komutunu uygulayın.Aşağıdaki uygulama konunun anlaşılmasına yardımcı olacaktır.

|  |
| --- |
| <html>  <head>  <title>switch Deyimi</title>  <script language="Javascript1.2">  function kontrol(form)  {  var sifre = form.sifre.value  switch(sifre)  {  case "Mehmet" :  alert("Iyi Günler! Mehmet Bey")  break  case "Cantekin" :  alert("Iyi Günler! Cantekin Bey")  break  default:  alert("Giris hakkiniz yok!!!")  }  }  </script>  </head>  <body>  <form>  <input type="text" name = "sifre">  <input type="button" value ="Tamam" onClick="kontrol(this.form)">  </form>  </body>  </html> |

Örnekte kullanıcının girdiği şifre, switch deyimindeki değerler ile karşılaştırıyor.Girilen değer,case lerde belirtilen değerlerden birine eşitse, ilgili case deyimi altındaki işlem yapılıyor. Hiçbir case değerine eşit olmadığı durumda ise default kısmında tanımlanan işlem yapılır.Verilen örneğin if deyimi ile yapılması da mümkündür; fakat anlaşılamsı kolay olsun diye böyle bir örnek verilmiştir.Daha fazlakarşılaştırmanın yapılması gereken durumlarda switch deyimi, hem programcının yazım için harcadığı süre hem de kullanıcının işlemler yapılırken beklediği süre açısından daha avantajlıdır.

Akış denetiminde kullanılan**continue** komutu; javascript’in döngünün o andaki adımını durdurup, döngünün başına dönmesini sağlar.Döngü baştan devam eder.

Örnek:

if(cevap="hayır")  
  {continue}

Ders 10: Döngü Denetimi

Javascript’te bazen işler açık ve belirgin şartlara bağlı ve bir kere icra edilecek türden olmayabilir.Bazen bir işin belirli kere yapmasını isteyebilirsiniz.Bu tür tekrarlanan işleri javascript’e bir kaç satırlık bir kodla yaptırmak, bizi çoğu kez sayfalar dolusu kod yazmaktan kurtarır.Bu nedenle dersimizde döngü denetimini ve döngü oluşturan döngü deyimlerinin üzerinde duracağız.

**Döngü deyimleri:**

* **for...**.
* **while**
* **do...while**dır.

**for Döngüsü**

For döngüsünün genel yapısı :

**for (** başlama noktası**;**koşul ifadesi**;**adım ifadesi**)**

**{**

İŞLEMLER

**}**

**for** döngüsünü belirleyen üç ifade vardır: **başlama noktası, koşul ifadesi, adım ifadesi**. Başlama noktası, döngünün hangi sayıdan başlayacağını belirler(i = 0 gibi).Koşul ifadesini daha önceki deyimlerden biliyoruz, koşul ifadesi aracılığıyla döngü devam eder(i <= 10 gibi). Adım ifadesi ise döngünün kaçar kaçar sayacağını belirleriz.Örneğin birer birer saymasını istiyorsak i++ değerini, ikişer ikişer saymasını istiyorsak i+=2 değerini kullanırız. Bu adımlar artırma (pozitif) yönlü olmak zorunda değil, azalma (negatif) yönlü de olabilir( i**--** gibi).

|  |
| --- |
| <html>  <head>  <title>for döngüsü</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var i  for (i=1; i<=10; ++i)  {  document.write("PAU \n")  }  </script>  </pre>  </body>  </html> |

For döngüsü i=1 değerinden başlıyarak, i<=10 şartını sağladığı sürece, birer birer artarak "PAU" yazısını yazacaktır."\n" her yazıştan sonra bir alt satıra inmesini sağlar.

Başka bir örnek daha yapalım.

|  |
| --- |
| <html>  <head>  <title>for döngüsü</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var i  for (i=100; i>=0; i -= 10)  {  document.write("Sayi = " + i + "\n")  }  </script>  </pre>  </body>  </html> |

Şimdi de html etiketlerini kullanarak tablolu bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>for döngüsü</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var i  document.write("<table border=\"1\">")  for (i=1; i<=10; i++)  {  document.write("<tr><td>" + i + "</td><td>Web Tasarimi</td></tr>")  }  document.write("</table>")  </script>  </pre>  </body>  </html> |

Örneğimizdeki tabloyu html etiketleri ile yazmak isteseydik; 10 tane satır, 20 tane hücre oluşturmamız gerekirdi.Ayrıca her hücrenin içini istediğimiz bilgiyle doldurmamız gerekirdi.Bunun yerini tekrar eden yerler for döngüsü içine alınarak, birçok kere yazma işinden kurtulduk.Böylece hem zamandan hemde kod kalabalığından kurdulduk.

**while Döngüsü**

**while** döngüsünde, koşul ifadesi sağlandığı sürece deyimdeki işlemler sürekli; bir döngü oluşturacak şekilde, kendisini tekrarlar. Bu işlemler öyle olmalıdır ki belli bir sayıda çevrimden sonra bu koşul ifadesini sağlamasın ve döngüden çıksın, aksi takdirde döngü sonsuza kadar devam eder.

while döngüsünün genel yapısı:

**while (**koşul ifadesi**)** işlem

birden fazla işlem söz konusuysa,

**while (**koşul ifadesi**)**

**{**

İŞLEMLER

**}**

Şimdi basit bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>while döngüsü</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var i =1  while(i < 8)  {  document.write("Sayi = " + i + "\n")  i++  }  </script>  </pre>  </body>  </html> |

while döngüsünde en çok yapılan hatalardan biri, şartta yazılı olan değişkenin değerini değiştirmeyi unutmaktır.Örneğimizde şart bir süre sonra sağlanmasın diye i değerini bir arttırdık.Bu sayede i değeri 8 olunca şart sağlanmadı ve döngüye girilmeden atlandı.

**do......while Döngüsü**

**do...while** döngüsü ile while döngüsü arasında işlevsel olarak fark yok gibi görünür, ancak do...while deyiminde koşul ifadesi altta bulunur; ve koşul değerlendirmesi işlemi çevrimden önce değil, sonra yapılır.Yani işlemler en az bir kere yapıldıktan sonra şarta bakılır.

do......while döngüsünün genel yapısı:

**do**

İŞLEMLER

**while (**koşul ifadesi**)**

birden fazla işlem söz konusuysa,

**do**

**{**

İŞLEMLER

**}**

**while (**koşul ifadesi**)**

İşleyişini daha iyi anlamak için bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>do while döngüsü</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var sayi = 1  do  {  document.write("Sayi degeri = " + sayi + "<br>")  sayi++  }while(sayi <=12)  </script>  </pre>  </body>  </html> |

Örnekte gördüğünüz gibi ilk önce do...while döngüsü içindeki işlemler yapılır, daha sonra şarta bakılır.Şart sağlandığı sürece döngü, başa döneceği için işlem, toplamda 12 kere yapılır.

Ders 11: Fonksiyonlar

Çoğu zaman sayfanızdaki bir javascript işleminin defalarca yapılması gerekebilir.Bu gibi durumlarda javascript'te başvurulan yöntem, fonksiyonlardır.Tekrarlanacak işlemler döngülerle de yapılabilir diye düşünebilirsiniz; fakat döngüler kullanılacakları yerde yazılırlar.Bir döngüyü ayrı ayrı yerlerde kullanmak isterseniz, tekrar tekrar yazmanız gerekir.Ayrıca fonksiyonlara bir takım değerler gönderip, fonksiyon bitiminde geriye değer döndürmekte mümkündür.Burada geriye sözüyle kastedilen yer, fonksiyonun çağırıldığı yerdir.Döngülerde ise böyle bir durum söz konusu değildir.

**Fonksiyon yapısı**

**function** fonksiyonun adı **(**argüman1,argüman2,....,argümanN**)**

**{**

İŞLEMLER

**}**

Fonksiyonlara koyduğumuz isim, fonksiyonu çağırmak için gereklidir.Her fonksiyonda argüman olması zorunlu değildir.Argüman, fonksiyonun girdi değeridir.Fonksiyonlarda yapılacak işlemler süslü parantezler içinde yazılır.

Fonksiyonlar <body>....</body> veya <head>....</head> etiketleri arasında yazılabilir; fakat tasarım düzgünlüğü ve performans açısından <head> etiketleri arasımda yazılması daha uygun olur.

Argümansız fonksiyon örneği

|  |
| --- |
| <html>  <head>  <title>Argumansiz Fonksiyon</title>  <script language="JavaScript1.2">  function mesaj()  {  alert("Hiç bir mesajiniz yok.")  document.write("Mesaja baktiniz...")  }  </script>  </head>  <body>  <form>  Mesaja bakmak istiyorsaniz, dügmeye basin.<br />  <input type="button" value="Mesaj Kontrol" onclick="mesaj()">  </form>  </body>  </html> |

Örneğimizde mesaj() isimli bir fonksiyon tanımladık.Tanımladığımız fonksiyonu kullanmak için formdaki butona eklediğimiz onclick olayını kullandık.Onclick olayı, tıklama ile çalışan bir olaydır.Bu olay ile ilgili detaylı bilgi ileride anlatacağımız olaylar konusunda yer alacaktır.

Argümanlı fonksiyon örneği

Daha önceki derslerimizde arasıra argümanlı fonksiyon örneği kullanılmıştı.Argümanlar fonksiyonlar, girdi değeri olarak ta adlandırııan argümanları alır ve fonksiyon içinde bu değerleri işlerler.Bunu bir örnek ile anlamaya çalışalım.

|  |
| --- |
| <html>  <head>  <title>Argumanli Fonksiyon</title>  <script language="JavaScript1.2">  function hesapla(form)  {  var s1 = parseInt(form.s1.value)  var s2 = parseInt(form.s2.value)  sonuc = s1 + s2  form.sonuc.value = sonuc  }  </script>  </head>  <body>  <form>  Toplama islemi için iki sayi giriniz<br>  1. sayi = <input type="text" name="s1"><br>  2. sayi = <input type="text" name="s2"><br>  <input type="button" value="Topla" onclick="hesapla(this.form)"><br>  Sonuç = <input type="text" name="sonuc">  </form>  </body>  </html> |

Örnekte kullanıcıdan iki sayı girmesi isteniyor.Girilen saylıar hesapla() isimli fonksiyonda toplanarak, sonuç yazdırılıyor.Yapılan işlemleri daha detaylı bir şekilde anlatmak gerekirse; girilen sayılar, onclick olayı ile fonksiyona gönderiliyor.Fonksiyon, formdaki değerleri argüman olarak alıyor ve fonksiyonda bu değerler toplanıyor.

Fonksiyonda kullanılan**parseInt()** metodu, girilen sayı değerlerini tamsayıya dönüştürülmesini sağlayan bir metottur.Eğer girilen değerler bu şekilde tamsayıya dönüştürülmezse, alfanümerik değer oldukları için ard arda eklenirler.Örneğin;1.sayı=15, 2.sayı=12 olursa, parseInt() metodu kullanılmadığı zaman, s1+s2 işleminin sonucu "1512" olur.

Son olarak ta bulunan işlem sonucu form içindeki "sonuc" isimli değişkenin değerine atanarak, fonksiyon bitiriliyor.

Dersimizin başında söylediğimiz gibi, fonksiyonlar bizi tekrarlanan işlemlerde, tekrar tekrar kod yazmaktan kurtarır.Yukarıda yaptığımız örnekle isteğiniz kadar toplama işlemi yapabilirsiniz.Hiçbir sınırlama yoktur.

**Değer Atamalı Fonksiyonlar**

Diğer bir fonksiyon yapısı da **değer atamalı fonksiyon**dur.Bu fonksiyon türü argümanlı veya argümansız olabilir.Ayırtedici özelliği yaptığı işlemleri bir değere dönüştürüp, bu değeri değişkenlere veya html nesnelerine aktarmasıdır.

Fonksiyonun yapısı

**function** fonksiyonun adı **(**argüman1,argüman2,....,argümanN**)**

**{**

İŞLEMLER  
**return** değer  
**}**

Bu sayede fonksiyonlarda sadece değerler üzerinde işlem yapmamış, aynı zamanda fonksiyon sonucunda bir değer üretmiş olursunuz.

Ders 12: Metodlar

Javascript, fonksiyonlara ve metodlara kendi nesnelerini ve onların metodlarını kullanma imkanı verir; hatta bilgisayarın kaynaklarından yararlanmalarını bile sağlar.Böylece javascript açısından browser ve onun görüntülediği html belgesinin nesne olarak özelliklerine ve bilgisayarın kaynaklarını kullanmaya biraz daha yaklaşılmış olunur.

**Zaman Metodları**

**Date()** metodu, javascript’in browserdan, browserın işletim sisteminden, işletim sisteminin de bilgisayarın temel girdi/çıktı işlemlerini yapan BIOS çipinden, saati ve tarihi içeren bilgiyi almasını sağlar.Date() metodu ile gün,ay, yıl, saat, dakika ve saniye bilgileri elde edilir.

**Zaman Değişkeni Tanımlama ve Yürürlükteki Zaman Değerini İfade Etme**

**new Date()**komutu ile temin edilen bilgiler, bir değişkene atanır.Daha sonrada "get" ifadesi ile başlayan komutlarla istenilen zaman bilgileri, değişken üzerinden temin edilerek kullanılır.

Zaman metodunda kullanılan kodlar

|  |  |
| --- | --- |
| getYear() | Yıl bilgisini verir. |
| getMonth() | Ay bilgisini verir.Ocak için 0, Şubat için 1, Mart için 2, ..., Aralık için 11 değerini verir. |
| getDate() | Gün bilgisini verir. |
| getDay() | Haftanın günlerini ifade eder. Pazar için 0, Pazartesi için 1, Salı için 2, ..., Cumartesi için 6 değerini verir. |
| getHours() | Saat değerini ifade eder. 24'lük sistem kullanılır.0-23 arası değer verir. |
| getMinutes() | Dakika bilgisini verir.0-59 arası değer verir. |
| getSeconds() | Saniye bilgisini verir.0-59 arası değer verir. |

Eğer bu kodların tanımlandığı değişken, önceden **var**değişken**= new Date()** şeklinde tanımlanmazsa kodlar çalışmaz; çünkü kodlar bu değişkene bağımlı olarak çalışır.Zaman bilgileri değişkene alınır, yukarıdaki kodlar ile istenilen zmana bilgileri değişkenden alınır.

Zaman metoduna ait kodları bir örnek üzerinde görelim.

|  |
| --- |
| <html>  <head>  <title>Zaman Metodlari</title>  </head>  <body>  <h4>Zaman Metodlari</h4>  <script language="Javascript1.2">  var zaman = new Date() //Zaman bilgileri bir degiskene atandi.  document.write("Zaman : " + zaman +"<br>")  document.write("getYear : " + zaman.getYear() +"<br>")  document.write("getDate: " + zaman.getDate() +"<br>")  document.write("getMonth : " + zaman.getMonth() +"<br>")  document.write("getHours " + zaman.getHours() +"<br>")  document.write("getMinutes : " + zaman.getMinutes() +"<br>")  document.write("getSeconds : " + zaman.getSeconds() +"<br>")  document.write("getDay: " + zaman.getDay() +"<br>")  </script>  </body>  </html> |

Örneği incelerseniz, uygulamayı çalıştırdığınız zamana ait bilgilerin görüntülendiğini göreceksiniz.Görülen değerler sadece uygulamanın çalıştırıldığı zamana ait değerlerdir.Eğer saat benzeri bir uygulama yapmak isterseniz, zamanı alma ve görüntüleme işlemlerini sürekli tekrarlamanız gerekir.

**Zaman Aralığı**

Belli zaman aralıklarıyla bir işlemin tekrarlanmasını isteniyorsa, **setTimeout(..., ...)** komutu kullanılır.İlk değer parantez içinde bir javaccript kodudur, ikinci değer ise ilk değerin ne kadar süre sonra oluşacağını belirleyen, mili saniye cinsinden zaman değeridir. (1 saniye = 1000 mili saniye)

setTimeout komutu daha çok zaman aralıklarıyla tekrar eden hareketler oluşturmak için kullanılır.

Örnek:

**setTimeout(**"alert('Bu mesaj 2 saniye sonra çıkar.')", 2000**)**

**Karakter Metodları**

Javascript'te sıkça kullanılan üç tane karakter metodu vardır.Bunlar:

* toString()
* toLowerCase()
* toUpperCase()        dir.

**tostring()** metodu; kelime anlamı olarak stringe(karaktere) çevir anlamına gelmektedir.Bu metodla, saat nesnesinin tamamen kendine özgü biçimi, javascript tarafından htmlin anlayabileceği şekle çevrilmiş olur.

Kullanım şekli:       değişken.toString()

Javascript’te, çevirme amaçlı iki metod daha vardır. toString() metodu sadece Date nesnesine uygulanabilir, ama toLowerCase() ve toUpperCase() metodları bütün alfanümerik değerlere uygulanabilir.

**toLowerCase()**    -> küçük harfe çevirmeye yarar.

**toUpperCase()**    - > büyük harfe çevirmeye yarar.

**Matematik Metodları**

Matematik metodlarından ilki **parseInt()**metodudur.Bu metod parentez içindeki değeri tamsayıya çevirmeye yarar.Diğer bir metod ise, **parseFloat()** metodudur.Bu metod ise, parantez içindeki değeri kesirli sayıya çevirir.parseInt() ve parseFloat() metodları formlarla ilgili uygulamalarda sıkça kullanılır.

Bunların dışında bir sürü matematik metodu vardır.Bunlar **Math.**ön komudu ile kullanılırlar.

* **round() :**Ondalık sayıları en yakın tam sayıya yuvarlar.

Kulanım şekli : Math.round(sayı)

* **ceil() :**Ondalık sayılar en yakında ve daha büyük olan tamsayıya eşitlenir.Tamsayılar için değişen birşey olmaz.round() ile farkı, round() kodunda ondalık sayının kendisinden düşük tamsayılara da eşitlenebilmesidir.

Kulanım şekli : Math.ceil(sayı)

* **floor():** Ondalık sayıyı en yakındaki ve daha küçük olan tamsayıya çevirir.

Kulanım şekli : Math.floor(sayı)

* **abs() :** Mutlak değer almaya yarar.

Kulanım şekli : Math.abs(sayı)

* **sqrt() :**Karekök alamay yarar.

Kulanım şekli : Math.sqrt(sayı)

|  |
| --- |
| <html>  <head>  <title>Matemetik Metodlari</title>  </head>  <body>  <pre>  <script language="Javascript1.2">  var s = 19.56  document.write("Sayi = " + s + "<br>")  document.write("Round = " + Math.round(s) + "<br>")  document.write("Ceil = " + Math.ceil(s) + "<br>")  document.write("Floor = " + Math.floor(s) + "<br>")  document.write("Abs = " + Math.abs(s) + "<br>")  document.write("Sqrt = " + Math.sqrt(s) + "<br>")  </script>  </pre>  </body>  </html> |

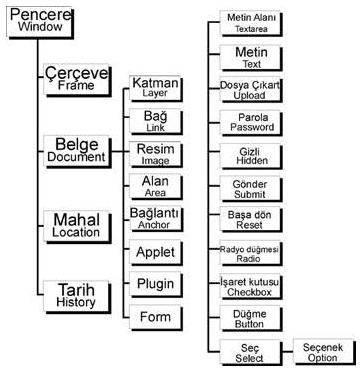
**Trigonometrik İşlemler**

Trigonometrik işlemler yapan matematik metodları aşağıda verilmiştir.Bunlarda Math. uzantılı kullanılırlar.

|  |  |
| --- | --- |
| **cos()** | Sayının kosinüsünü gösterir. (radyal değerler kullanın) |
|  | Math.cos(3.141592653589793) |
| **sin()** | Sayının sinüsünü gösterir |
|  | Math.sin(3.141592653589793/2) |
| **tan()** | Sayının tanjantını gösterir. |
|  | Math.tan(3.141592653589793/4) |
| **acos()** | Sayının arkkosinüsünü gösterir. (radyal değerler verir.) |
|  | Math.acos(0.5) |
| **asin()** | Sayının arksinüsünü gösterir. |
|  | Math.asin(0.25) |
| **atan()** | Sayının arktanjantını gösterir. |
|  | Math.atan(1) |

Ders 13: Nesneler

Javascript ile yazılan programlar, browser programlarının belge nesne modeli denen kurallar içinde hareket etmek zorundadır.Javascripti, browserın sunduğu hiyerarşik nesne modeli ile sınırlıdır. Aşağıda Netscape’in belge nesne modeli verilmiştir.Internet Explorer çok daha fazla nesneyi kontrol etmenize imkan verir; fakat derslerimizde her browserda algılanacak örneklere yer verdiğimiz için bu nesne modeli verilmiştir.



Önceki derslerimizde ekrana bir yazı yazdırmak istediğimizde bunu document.write komutu ile yapmıştık.Bu işlemde şöyle bir mantık vardır.Biz, pencerenin içindeyiz, hitap edeceğimiz nesne ise döküman, öyleyse yazmaya hitap edeceğimiz nesnenin bulunduğu kademeden başlamalıyız.

Nesnelerin kullanım mantığında aşamalılık vardır.Bulunduğun düzeyden başlarak, hitap edeceğin nesne düzeyine kadar belirtilmesi gerekir.Daha önceki derslerimizden hatırlayacağınız gibi, herhangi bir form aracıyla işlem yapacağımızda form.araççubuğunun\_adi.value demiştik.Aslında biz pencerenin içinde bulunduğumuz için yazmaya döküman nesnesinden başlamalıyız.document.form.araççubuğunun\_adi.value şeklinde tanımlanmalı; fakat javascript kurallar yönünden esnek bir dil olmamasına rağmen, bu kullanımda bie seneklik sağlamıştır.

Şemadaki nesnelerin html karşılıkları :

|  |  |
| --- | --- |
| Nesneler | Html Karşılıkları |
| Pencere (Window) | Yok |
| Çerçeve (Frame) | <frame> |
| Belge (Document) | <body> |
| Form | <form> |
| Düğme(button) | <input type="button"> |
| İşaret Kutusu | <input type="checkbox"> |
| Gizli (Hidden) | <input type="hidden"> |
| Dosya Çıkart | <input type="file"> |
| Parola (password) | <input type="password"> |
| Radyo düğmesi (Radio) | <input type="radio"> |
| Başa dön (Reset) | <input type="reset"> |
| Seç (Select) | <select> |
| Gönder (Submit) | <input type="submit"> |
| Metin (Text) | <input type="text"> |
| Metin Alanı (Textarea) | <textarea> |
| Bağ (Link) | <a href="xxx" > |
| Bağlantı (Anchor) | <a name="xxx"> |
| Resim (Image) | <img src="xxx"> |
| Plugin <embed> | <embed rsc="xxx"> |
| Alan (Area) | <map> |
| Applet | <applet> |

Pencere(window) nesnesin html etiketi olarak karşılığı yoktur; ama değişik uygulamalarla pencere nesneleri kullanılabilir.Şimdi böyle bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>Pencere Nesnesi</title>  <script language="Javascript1.2">  var pencereNesnesi  function yenipencere()  {  pencereNesnesi = window.open("", "YeniPencere", "width=300, height=150, resizable=1")  pencereNesnesi.document.write("Yeni bir pencere açtiniz..")  }  </script>  <body>  Dügmeyi tiklarsaniz, yeni pencere açilir.  <form>  <input type="button" value="Yeni Pencere Aç" onClick="yenipencere()">  </form>  </body>  </html> |

Pencere nesnesi ile ayrıntılı bilgi sonraki derslerimizde verilecektir.Burada kısaca, pencere nesnesine ait html etiketi olmamasına karşın, bu nesnelerin değişik tanımlamalar ile kullanıldığı gösterildi.Pencere nesnesine atıfta bulanabilmek için, window.open işlemi bir değişkene atandı.Daha sonra bu değişken ismi kullanılarak yeni açılan pencereye yazı yazıldı.

Pencere açma işleminin window.open() ile yapıldığını gördük.Penceryi kapatmak için ise window.close() kullanılır.İsterseniz window.status = "........." ile penceresinin alt çervesindeki durum alanına yazı da yazdırabilirsiniz.

Ders 14: Olaylar

Javascript'te yapılan bir çok şeyde olay kullanılır.Kullanıcının sayfa üzerinde yaptığı her türlü işlem olaydır.Sayfayı açmak, bir nesnenin üzerine tıklamak, fareyle bir nesnenin üzerine gelmek, bir tuşa basmak, bir değeri değiştirmek.Bunların hepsi bir olaydır.Şimdi javascript'te sıklıkla kullanılan olaylardan bahsedelim.

**onClick olayı :**

Bu derse kadar **onclick** (tıklanıldığında) olayını düğme üzerinde birçok uygulamamızda kullandık.Bu olayı uygulamak için mutlaka düğme kullanılacak diye bir şart yoktur, herhangi bir html nesnesi de olabilir.onClick olayı; button, reset, submit, radio, checkbox ve area ile kullanılabilir.

|  |
| --- |
| <html>  <head>  <title>onClick Olayi</title>  <script language="Javascript1.2">  function mesaj()  {  sayfa = window.open("","yeni\_sayfa", "toolbar=1,width=320, height=80")  sayfa.document.write("<h1>Merhaba! Yeni bir sayfa...</h1>")  }  </script>  </head>  <body>  <a href="#" onClick="mesaj()">Mesajiniz var.</a>  </body>  </html> |

**onDblClick olayı :**

onclick olayına çok benzeyen diğer olay: **ondblclick** (çift tıklanıldığında)tir.Adından da anlaşılacağı gibi kullanıcının fareyle çift tıklamasını ifade eder.Oldukça alışılmış bir olaydır.onDblClick olayının yarattığı etkiyi görmek için yukarıdaki örnekte onClick yerine onDblClick yazın.

**onSubmit olayı :**

Javascript’in, htm i tamamladığı noktaların başında form ve form yoluyla elde edilecek bilgilerle ilgili işlemler gelir.Bu işlemlerin başında ise formdaki bilgilerin server a gönderilmeden önce, daha browserda iken doğrulanması ve hatalardan arındırılması vardır.Javascript, bunu ziyaretçinin formda, submit(gönder) olayına yol açan herhangi bir düğmenin tıklanması halinde yapar.Bunu şimdi yapacağımız örnekle daha iyi anlayacaksınız.

|  |
| --- |
| <html>  <head>  <title>onSubmit Olayi</title>  <script language="Javascript1.2">  function onay()  {  return confirm("Formu gönderiliyor.Devam etmek istermisiniz?")  }  </script>  </head>  <body>  <form method="get" onSubmit="return onay()">  Isim : <input type="text" name="isim">  <input type="submit" value="Gönder">  </form>  </body>  </html> |

<form method="get" **onSubmit="return onay()"**> olayı ile form, gönderilmeden önce bir fonksiyona gönderiliyor.return kelimesi ile fonksiyondan değer geleceği anlaşılıyor.onay() fonksiyonunu incelersek confirm komudu ile üzerinde tamam ve iptal düğmelerinin bulunduğu bir pencere açılıyor.Tamam düğmesine basıldığında form bilgisi gönderiliyor.Bunu adres çubuğuna bakara ta görebilirsiniz.Eğer iptal düğmesine basılırsa kullanıaln return deyimlerinden dolayı, forma geri dönülüyor.

**onReset olayı :**

Formlara koyulan reset(başa dön) düğmesi, bir formdaki bütün bilgilerin silinmesini sağlar. Browserın geri düğmesini tıklamak dahil, hiç bir eylem ziyaretçinin yanlışlıkla bastığı sil düğmesinin sildiğini geri getiremez.Fakat javascript ile böyle bir hata önlenebilir.Çünkü sil düğmesinin oluşturduğu "reset" olayını yönlendiren onReset, bize bu imkanı verir.Bunu bir örnekle ifade edelim.

|  |
| --- |
| <html>  <head>  <title>onReset Olayi</title>  <script language="Javascript1.2">  function kurtar()  {  return confirm("Bilgiler silinecektir.Devam etmek istermisiniz?")  }  </script>  </head>  <body>  <form method="get" onReset="return kurtar()">  Isim : <input type="text" name="isim">  <input type="reset" value="Sil">  </form>  </body>  </html> |

**onChange olayı :**

Html sayfasında değişiklik, ziyaretçinin içine bir şeyler yazabildiği veya bazı işaretler koyabildiği üç alandır.Bunlar : Text (Metin), Textarea (Metinalanı) ve Select (Seç) etiketleridir. Kullanıcı klavyeye basarak, bu alanlara metin girebilir.Bu durum html sayfasında değişiklik olması anlamına gelir.Select etiketinin sağladığı şıklardan birini seçen kullanıcı da sayfada değişiklik olayına yol açar.

**onFocus ve onBlur olayları :**

Html sayfasında da browserın focusunu(dikkat odağını) üzerinde topladığı veya odağın çekildiği üç nesne olabilir.Bunlar: Text (Metin), Textarea (Metinalanı) ve Select (Seç) etiketleridir.Çünkü ziyaretçinin sadece bu etiketlerin oluşturduğu nesnelere klavyenin dikkatini çekme hakkı vardır.Bu nesnelerden biri tıklanınca browserın dikkati bu nesneye dönmüş, yani "focus" olayı olmuş demektir.Bu durumda, biz de bu olayı yönlendiren **onFocus** u kullanabiliriz. Aynı mantıkla, ziyaretçi browserın dikkatini bu nesneden başka bir yere çektiği zaman bu nesne focus’u kaybeder, blur(netlikten çıkma, flulaşma) olayı olur. Bunu da **onBlur** ile yönlendiririz. Şimdi bu olayların kullanıldığı bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>onFocus ve onBlur olaylari</title>  <script language="Javascript1.2">  function goster()  {  document.isimForm.ad.value="Adiniz kayda geçmistir."  }  function uyar()  {  document.isimForm.ad.value="Lütfen yanlislik yapmayiniz!"  }  </script>  </head>  <body>  <form name="isimForm">  <p>Lütfen adinizi yazin ve sayfa üzerinde baska bir yeri tiklayin:  <br><input type="text" name="ad" value="Adinizi buraya yaziniz" size=25 onBlur="goster()">  <p>Simdi de asagidaki kutuya mesleginizi yazin:  <br><input type="text" name="meslek" value="Mesleginizi buraya yaziniz" size=25 onFocus="uyar()">  </form>  </body>  </html> |

Javascript, üstteki kutuya adınızı yazdıktan sonra başka bir yeri tıkladığınızda olan iş, "ad" isimli metin kutusunun "blur" olayına onBlur’un çağırdığı goster() fonksiyonu ile; alttaki kutuyu mesleğinizi yazmak üzere tıkladığınız anda "meslek" kutusunda olan "focus" olayına da uyar() fonksiyonu ile karşılık veriyor.

**onMouseOver ve onMouseOut olayları :**

**onMouseOver** olayı, mouse imlecinin bir html nesnesi üzerine gelmesini ifade eder.   
**onMouseOut** olayı ise , mouse imlecinin bir html nesnesi üzerinden gitmesini ifade eder.

Şimdi bu olayları güzel bir örnekle gösterelim.

|  |
| --- |
| <html>  <head>  <title>onMouseOver ve onMouseOut Olaylari</title>  <script language="Javascript1.2">  function goster()  {  alert("Yazinin üzerine geldiniz..")  }  function mesaj()  {  alert("Yazinin üzerinden gittiniz..")  }  </script>  </head>  <body>  <a href="#" onMouseOver="goster()">Mesaj için yazinin üzerine gelin..</a><br><br><br>  <a href="#" onMouseOut="mesaj()">Mesaj için yazinin üzerinden gidin..</a>  </body>  </html> |

**onLoad ve onUnLoad olayları :**

Javascript açısından browserın bir html sayfasını görüntülemeyi bitirmesi sayfanın load (yükleme), başka bir sayfaya geçilmesi ise önceki sayfanın unload(yüklenmişlikten çıkması) sayılır ve bu iki olayı **onLoad** ve **onUnLoad** yönlendiricileri ile karşılayabiliriz.Browser açısından asıl sayfa, body bölümüdür.Head bölümündeki komutlar, sayfanın nasıl görüntüleneceğine ilişkin komutlar içerir.Bizde şimdi body etiketini kullnarak bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>onLoad ve onUnLoad Olaylari</title>  <script language="Javascript1.2">  function hosgeldin()  {  alert("Hosgeldiniz...!")  }  function gule()  {  alert("Güle güle! Yine Bekleriz.")  }  </script>  </head>  <body onLoad="hosgeldin()" onUnLoad="gule()">  </body>  </html> |

Javascript'te sıklıkla kullanılan olaylar bu kadardır.Sizde değişik örnekler yaparak olaylar konusunu daha iyi anlayabilirsiniz.

Ders 15: Browser Nesneleri - Pencere (windows)

Browserın javascript dilindeki adı Navigator’dır.Ziyaretçinin Browser programı ne olursa olsun, javascript için navigatordır.Nesnelerine de navigator nesneleri denir.Bunların başında, browser’ın pencereleri gelir.

Şimdiki derse kadar bazı derslerimizde javascript kodunda navigator(browser) penceresini ve bazı özelliklerini kullandık.Şimdi pencere özelliklerinin hepsini verelim.

Pencere(window) özellikleri:

|  |  |
| --- | --- |
| width | Navigator’ın sayfanın görüntülenmesi için ekrandaki temiz alanının piksel olarak genişliği. |
| height | Navigator’ın sayfanın görüntülenmesi için ekrandaki temiz alanının piksel olarak yüksekliği. |
| toolbar | Navigator’ın araç çubuğunun gösterilmesi (=1) veya gizlenmesi (=0). |
| menubar | Navigator’ın menü çubuğunun gösterilmesi (=1) veya gizlenmesi (=0). |
| scrollbars | Navigator’ın sağ ve alt kenardaki kaydırma çubuklarının gösterilmesi (=1) veya gizlenmesi (=0). |
| resizable | Navigator penceresinin büyütülebilir-küçültülebilir olması (=1) veya olmaması (=0). |
| status | Navigator penceresinin alt kenarındaki mesaj çubuğunun gösterilmesi (=1) veya gizlenmesi (=0). |
| location | Navigator penceresinin URL adres çubunun gösterilmesi (=1) veya gizlenmesi (=0). |
| directories | Netscape’de ikinci araç çubunun gösterilmesi (=1) veya gizlenmesi (=0). |
| copyhistory | Mevcut Navigator penceresinin history kaydının (daha önce ziyaret edilen URl adreslerinin tutulduğu çizelgenin) yeni pencereye de kopyalanması. |
| outerWidth | Navigator penceresinin piksel olarak genişliği. |
| outerHeight | Navigator penceresinin piksel olarak yüksekliği. |
| left | Navigator penceresinin ekranın sol kenarından piksel olarak uzaklığı. |
| top | Navigator penceresinin ekranın üst kenarından piksel olarak uzaklığı. |
| alwaysRaised | Navigator penceresinin masaüstünde açık bütün pencerelerin üzerinde kalmasını sağlar. (Sadece Windows ve MacOS’de işler.) |
| z-lock | Navigator penceresinin içi tıklansa bile masaüstünde açık bütün pencerelerin altında kalmasını sağlar. (Sadece Windows ve MacOS’de işler.) |

1, evet ve 0, hayır anlamına gelmektedir.Bunların yerine yes (evet) ve no (hayır) da kullanılabilir.

Javascriptin yeni navigator penceresi açma metodu, **window.open()** şöyledir:

pencerenesnesi = window.open("sayfa\_adresi", "YeniPencereninAdı", "pencere\_özellikleri")

"sayfa\_adresi"nin kullanılması zorunlu değildir.Açılacak pencerede önceden varolan bir sayfanın açılması istendiğinde adresi ve adı burada belirtilir.İstenmezse çift tırnak işaretlerinin içi boş bırakılır.Pencereye ait tüm özellikler çift tırnaklar içinde içinde belirtilir.Kullanımda yer alan "pencerenesnesi" şeklinde belirtilen unsur ise pencerenin kapatılması gibi işlemlerde pencereye hitap etmek için kullanılır.

Pencereyi kapatmak için **xxx.close()** metodu kullanılır."xxx" yerine pencereye hitap etmek için kullanılan isim yazılır.Eğer isim belirtilmezse browser komutlarının icra edildiği pencere kapatılır.

Şimdi bunları bir örnekle görelim.

|  |
| --- |
| <html>  <head>  <title>Pencere(window)</title>  <script language="Javascript1.2">  function ac()  {  ypencere = window.open("","yenipencere", "toolbar=1, status=1, width=475, height=80")  ypencere.document.write("Sol alt kösedeki durum çubuguna bakin.")  ypencere.window.status="Durum çubuguna mesaj yazabilirsiniz."  }  function kapat()  {  ypencere.close()  }  </script>  </head>  <body>  <input type="button" value="Pencere Aç" onClick="ac()">  <br><br><br>  <input type="button" value="Pencereyi Kapat" onClick="kapat()">  </body>  </html> |

Örneğimizde pencere açma ve kapatma olayları fonksiyonlar ile yapıldı.Açılan pencerede üzerinde işlemler yaparken ve bu pencereyi kapatırken, pencereye hitap etmek için verdiğimiz ismi kullandığımıza dikkat edin.

Örnekte **window.status = ".........."** komutu kullanıldı.Bu komut javascript'te durum çubuğuna yazı yazdırmak için kullanılır.Bizde yenia açılan pencernin durum çubuğuna yazı yadırmak için bu komuttan yararlandık.

Browser kendisine ait bazı özellikler nedeni ile javascript açısından nesne olarak algılanır.Bu açıdan browserın javascript için şu özellikleri vardır:

|  |  |
| --- | --- |
| appname | Browser’ın adı |
| appVersion | Browser’ın sürümü |
| appCodeName | Browser’ın kod adı |
| userAgent | Browser’ın Server’a kendisini tanıtırken verdiği isim |

Şimdi bunları bir örnek üzerinde görelim.

|  |
| --- |
| <html>  <head>  <title>Browseri Tanyalim</title>  <pre>  <script language="Javascript1.2">  function browsertanit ()  {  document.write( "Browser: " + navigator.appName + "<br>")  document.write( "Sürümü:" + navigator.appVersion + "<br>")  document.write("Kod Adi: " + navigator.appCodeName + "<br>")  document.write( "userAgent: " + navigator.userAgent + "<br>")  }  </script>  </pre>  </head>  <body>  <input type="button" value="Browser hakkinda bilgi !" onClick="browsertanit()">  </body>  </html> |

Örnek ile kullandığınız browsera ait özellikleri göreceksiniz.

Ders 16: Browser Nesneleri – Mesaj Kutuları

Navigator penceresi, üç tür mesaj kutusu açabilir. Bunlar:

* alert     -> Basit bilgi verme kutusu
* confirm -> Doğrulama kutusu
* prompt -> kKullanıcının bilgi girme kutusudur.

**alert(......)**

Uyarı kutusu yapmaya yarayan bu kodu daha önceki derslerimizde de kullanmıştı. JavaScript değişkenlerini ve tırnak içinde olmak şartıyla html kodlarını uyarı kutusuyla gösterebiliriz ve **"+"**işareti ile birleştirilir.

|  |
| --- |
| <html>  <head>  <title>Alert</title>  <script language="Javascript1.2">  function kare()  {  var s  s = 16 \* 16  alert("16 sayisinin karesi = " + s)  }  </script>  </head>  <body>  <form>  16 sayisinin karesini ögrenmek istiyorsaniz dügmeye basin.<br>  <input type="button" onClick="kare()" value="Sayinin karesini bul!">  </form>  </body>  </html> |

**confirm(......)**

Bu mesaj kutusu ile kullanıcının onayı alınır. Bu kutuyu yaratmak için **confirm(...)** komutu bir değişkene eşitlenir.Kullanıcının onayı true, onay vermemesi ise false şeklinde ifadelendirilir.

|  |
| --- |
| <html>  <head>  <title>Confirm</title>  <script language="Javascript1.2">  function onayla()  {  var onay  onay = confirm("Yapilacak isleme onay vermek istermisiniz?")  if (onay == true)  document.write("Onay verdiginiz için tesekkürler!")  else  document.write("Onay vermediniz! Islemler iptal edildi.")  }  </script>  </head>  <body>  <form>  Mesaj kutusundaki iki dügmeyide deneyerek, olusan durumlari gözlemleyiniz.<br>  <input type="button" value="Onay" onClick="onayla()">  </form>  </body>  </html> |

**prompt(......)**

**prompt**sayesinde kullanıcıyla program arasında bir veri alışverişi yapılır.prompt, tıpkı confirm gibi bir değişkene eşitlenir.prompt komutu iki değer alır. İlk değer yazı kutusunda yer alan bilgi verici yazıyı belirler, ikinci değer ise yazı kutusunun yazma yerinde yer alan yazıyı belirler, bu yazı genellikle girdi bilgisi içerir.Kullanıcının yazdığı yazı, prompt komutunun eşitlendiği değişkene değer olarak atanır.Yazı kutusunda iptal düğmesi yer almaktadır. Kullanıcı bu düğmeye bastığında değişken null değeri alacaktır.

|  |
| --- |
| <html>  <head>  <title>Prompt</title>  <script language="Javascript1.2">  function kabul()  {  alert("Dogru sifre! Hosgeldiniz!")  }  function red()  {  alert("Sifreyi ögrendikten sonra deneyin!")  }  </script>  </head>  <body>  <script language="Javascript1.2">  var sifre = prompt("Sayfaya açabilmek için sifreyi giriniz!--Sifre=pau--","Sifre girisi...")  if (sifre == "pau")  kabul()  else  red()  </script>  </body>  </html> |

Ders 17: Browser Nesneleri – Çerçeve Nesnesi

Çerçevelerin, navigator açısından nasıl bir hiyerarşi işlediğini nesneler dersinde vermiştik. Javascript açısından her bir çerçeve bir pencere sayılır.Bunlara atıfta bulunurken şu adlandırma uygulanır:

|  |  |
| --- | --- |
| top | En üst pencere. yani Browser’ın kendisi. |
| parent | Herhangi bir frame i oluşturan Frameset.Çerçeve çerçeve içinde ise, bir çerçevenin içinde bulunduğu çerçeve parent sayılır. Sayfada bir Frameset varsa, bütün çerçeveler için "top" aynı zamanda "parent" sayılır. |
| self | Çerçevenin kendisi. |

Javascript, bir parent Frameset’in yavru çerçevelerini 0’dan itibaren numaralar.Yani, sayfanızda iki çerçeveniz varsa, birincisi "parent.frames[0]" ikincisi ise "parent.frames[1]" adıyla bilinir.

Şimdi adım adım bir örnek yapalım.

İki çerçeveli bir frameset için aşağıdaki kodlar anacerceve.asp ismi ile kaydedildi.

|  |
| --- |
| <html> <head> <title>Dinamik Cerceve</title> </head> <frameset cols="\*,\*"> <frame src="jsd17nin/ic1.asp" name="cerceve1"> <frame src="jsd17nin/ic2.asp" name="cerceve2"> </frameset> <noframes> <body> </body> </noframes> </html> |

Şimdi ise çecevelerde yer alacak sayfaları hazırlayalım.   
ic1.asp sayfasının içeriği:

|  |
| --- |
| <html> <head> <title>ic1</title> <script LANGUAGE="JavaScript1.2"> function renklen() { parent.frames[1].document.clear(); parent.frames[1].document.write("<html><head>" + "<title>Dinamik Güncelleme Örnegi</title>"); parent.frames[1].document.write(" </head><body bgcolor=\"" + document.form1.input1.value + "\">"); parent.frames[1].document.write("<h2>Renklendirilmiş sayfa</h2>"); parent.frames[1].document.write("</body></html>"); parent.frames[1].document.close(); } </script> </head> <body> <form NAME="form1"> Metin kutusuna ingilizcebir renk ismi veya renk kodu giriniz.<br><br> <input type="text" name="input1" size="20"><br> <input type="button" value="Yandaki çerçeveye renk verelim" onClick="renklen()"> </form> </body> </html> |

ic2.asp sayfasının içeriği:

|  |
| --- |
| <html> <head> <title>ic2</title> </head> <body> <h3>Bu sayfa renklendirilecek</h3> </body> </html> |

Örneğimizde, document.write() metoduyla, hedef çerçevemiz olan iki numaralı çerçevenin (parent.frames[1]) birinci çerçevedeki formdan aldığımız unsurları kullanarak, zemin rengini ve içeriğini değiştirdik.Sizde bu örnekten yola çıkarak, bir çerçevede meydana gelen değişikliklerden yola çıkarak, diğer çerçevelerde değişimler meydana getirebilirsiniz.

Ders 18: Belge (Döküman) Nesneleri

Döküman (document) nesnesi html dokümanının tamamını ifade etmektedir ve html de kullandığımız <p>, <h1>, <div>, vb. gibi tüm komutları içerir.Kod yazarken döküman nesnesini htmldeki <body> bölümü olarak düşünebilirsiniz.

Belge nesnelerinde önemli noktalardan bir aşamalılıktır.Örneğin, form içindeki "isim" adlı bir metin kutusuna hitap edeceksek; document.form.isim.value tanımlamsı kullanılmalıdır. Javascript'in belge nesnesi iç içe kaplara benzer ve tanımlamalarda bu şekilde yapılmalıdır.Şimdi kısaca sık kullanılan döküman nesnelerinden bahsedelim.

* **document.write() :**Sayfa içinde yazı yazdırmak için kullanılır.

Örnek : document.write("Merhaba")

* **document.writeln() :** Sayfa içine yazı yazdırmak için kullanılır.Bunun document.write() tan farkı yazım işileminden sonra bir alt satıra geçmesidir.Javascript tarafından document.writeln() komutunun algılanması için javascript kodlarının <pre> etiketleri içine yazılması gerekir.

Örnek : document.writeln("Merhaba")

* **document.title :**Sayfanın başlığını değiştirmek veya sayfaya başlık vermek için kullanılır.

Örnek : document.title = "Belge Nesneleri Dersi"

* **document.bgcolor :**Arkaplan rengini değiştirmeye yarar.

Örnek : document.bgcolor = "blue"

Daha önceki derslerimizde belge nesnelerini sıkça kullandık.Şimdi kullanıcının sayfa başlığını değiştirebileceği basit bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>Sayfa basligini degistirme</title>  <script language="Javascript1.2">  function degistir(form)  {  var baslik = form.baslik.value  document.title = baslik  }  </script>  </head>  <body>  Metin kutusuna sayfa basligi olmasini istediginiz metni yazarak, tamam butonuna basiniz.Sayfa basliginin degistigini göreceksiniz.  <form>  <input type="text" name="baslik"><br>  <input type="button" value="Tamam" onClick="degistir(this.form)">  </body>  </html> |

* **innerHTML :**Döküman nesnenin içeriğini değiştirmek için kullanılır.Önce döküman **id="..."**parametresi ile tanımlanır.Tanımlanan döküman "dökümantanımı*.*innerHTML" ile içeriği değiştirir.

|  |
| --- |
| <html>  <head>  <title>innerHTML Ornegi</title>  <script language="Javascript1.2">  function cikis()  {  degis.innerHTML="Yazinin degistigini fark ettiniz mi?"  }  function giris()  {  degis.innerHTML="Yazinin üzerine gelince, degisir."  }  </script>  </head>  <body>  <h1 id="degis" onMouseOver="cikis()" onMouseOut="giris()" >Yazinin üzerine gelince, degisir.</h1>  </body>  </html> |

Ders 19: Form Nesneleri

Javascript’in en önemli özelliklerinden biri ziyaertçi ile etkileşim kurma olanağı vermesidir. Ziyaretçi size bilgileri ancak ve sadece form ile ulaştırabilir. Bu sebeple javascript açısından html in en önemli nesneleri, form nesneleridir.

Form html etiketleri ile oluşturulur.Bu bakımdan javascript e ihtiyaç yoktur.Javascript formdaki bilgilerin işlenmesi sırasında devreye girer.

Sayfadaki form nesnelerini, **name** parametresiyle tanımladığımız isimler aracılığıyla çağırabiliriz.Örneğin, document.formbilgileri gibi.Form komutlarını da bu kök üzerine yerleştiririz.

Bilgi giriş elemanlarının (düğme, metin kutusu, kontrol kutusu, radio vb.) yapısı ve parametreleri html derslerinde anlatmıştı.İstersek o parametreleri javaScript koduyla tanımlayabiliriz.Örneğin metin kutusunun uzunluğunu karakter cinsinden belirleyen **size** parametresini javaScript kodlarıyla tanımlayan bir örnek ile bunu görelim.

|  |
| --- |
| <html>  <head>  <title>Form Nesneleri</title>  <script language="Javascript1.2">  function uzunluk(x)  {  document.Form1.mesaj.size=x  }  </script>  </head>  <body>  <form name="Form1">  Metin :<input name="mesaj"><br><br><br>  <input type="button" onClick="uzunluk(10)" value="Kisalsin">  <input type="button" onClick="uzunluk(70)" value="Uzasin">  </form>  </body>  </html> |

Diğer parametreleri de bu şekilde kullanabilirsiniz.Örneğin maksimum karakter girişini belirlemek için maxlength kodundan yararlanabiliriz.Örnek:document.Form1.mesaj.maxlength

Form araçlarında sıkça kullanılan**value** parametresi vardı.Bu parametre ile metin kutusuna yazı yazdırıyor yada düğmelere isim veriyorduk.JavaScript kodlarıyla bu parametreyi, metin kutusuna yazılı olan ifadeyi değişkenlere aktarmak için de kullanabiliriz.Kısacası metin kutusundaki ifadeyi value parametresi ile ifade ederiz.Bu değeri okurken veya değiştirirken value parametresinden faydalanırız.

Şimdi sizinde hoşunuza gidecek bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>Form Nesneleri-value</title>  <script language="Javascript1.2">  function girBilgi(form)  {  bilgi1.innerHTML=document.kimlik.veri1.value  bilgi2.innerHTML=document.kimlik.veri2.value  bilgi3.innerHTML=document.kimlik.veri3.value  }  </script>  </head>  <body>  <form name="kimlik">  Adiniz:<input name="veri1"><br>  Soyadiniz:<input name="veri2"><br>  Yasiniz:<input name="veri3">  <input type="button" value="Bilgileri Tabloya gir" onClick="girBilgi(this.form)">  <br><br>  <table cellpadding="5" border="1">  <tr><td width="100">Adi:</td><td id="bilgi1" width="300"> </td></tr>  <tr><td>Soyadi:</td><td id="bilgi2"> </td></tr>  <tr><td>Yasi:</td><td id="bilgi3"> </td></tr>  </table>  </form>  </body>  </html> |

Örnekte basit bir bilgi giriş formu yapıldı.Fakat, girilen bilgiler pencere kapatılınca yok oluyor. Buradan yola çıkarak şunu şöyleyebiliriz; javascript, bilgileri aktarmaya ve dönüştürmeye yarar ancak bilgileri saklamaya yaramaz.Bunun için veri tabanı gereklidir.

Kontrol kutularını seçmek için **checked** parametresini true değeri verebiliriz veya bunu JavaScript ile "document.form1.kontrolkutusu1.checked=true" şeklinde ifade edebiliriz. Radyo kutusunun işlevi gereği sadece birinin seçilmesi için, her kutuya aynı isim veriliyordu.Bu durumda yukarıda yazıldığı gibi name özelliğini kullanamayız ama id parametresini kullanarak bu sorunu halledebiliriz. Bunu size bir örnek ile açıklayalım.

|  |
| --- |
| <html>  <head>  <title>Form Nesneleri</title>  <script language="Javascript1.2">  function onayla(form)  {  if (document.soru.rad1.checked==true) secim.innerHTML="Seçtiginiz seçenek : Bilgisayar"  if (document.soru.rad2.checked==true) secim.innerHTML="Seçtiginiz seçenek : Elektronik"  if (document.soru.rad3.checked==true) secim.innerHTML="Seçtiginiz seçenek : Makina"  }  </script>  </head>  <body>  Asagidaki kutulardan birini seçikten sonra onaylayiniz.<br>  <form name="soru">  <input type="radio" name="rad" id="rad1">Bilgisayar<br>  <input type="radio" name="rad" id="rad2">Elektronik<br>  <input type="radio" name="rad" id="rad3">Makina<br>  <input type="button" onClick="onayla(this.form)" value="Onay">  <h4 id="secim">Yaptiginiz seçimin sonucu burada görüntülenecek.</h4>  </form>  </body>  </html> |

Örnekte if(document.soru.rad1.checked==true) yerine if(document.soru.rad1.checked) şekline bir kullanımda yapılabilir.Javascript her ikisi içinde aynı sonucu veir.

Radyo kutuları üzerinde başka türlü bir çalışma yöntemi daha vardır.Bu yöntemde, radyo kutularında "id" leri değil "value" ları kullanırsınız.Hangi radyo kutusunun seçili olduğunu anlamak için ise, "**radyoKutusu[i].checked**" yapısını kullanabilirsiniz.

 Html derslerinde seçenek kutusu ile ilgili bilgileri vermiştik şimdi ise seçenek kutusu ile ilgili uygulamaların nasıl yapıldığına bakalım.Seçenek kutusunda kullanıcının işaretlediği seçeneği **selectedIndex** komutuyla indeks (sıra) numarası cinsinden ifade ederiz.Şimdi basit örnek yapalım ve açıklamalarımıza sonra devam edelim.

|  |
| --- |
| <html>  <head>  <title>Form Nesneleri</title>  <script language="Javascript1.2">  function secim(form)  {  var i=document.sira\_no.bolum.selectedIndex  alert((i+1) + ".inci seçenegi seçtiniz")  }  </script>  </head>  <body>  <form name="sira\_no">  <select name="bolum">  <option>Bilgisayar  <option>Elektronik  <option>Otomativ  <option>Makina  <option>Insaat  </select>  <input type="button" value="Kaçinci seçenegi seçtigimi göster" onClick="secim()">  </form>  </body>  </html> |

Bu örnek, kullanıcının seçtiği seçeneğin sıra numarasını gösterir.selectedIndex komutu seçeneklere 0 (sıfır)dan başlayarak değer verdiği için, örnekte sıra numarasını i+1 şeklinde ifade ederek yazdırdık.

Seçenek kutusundaki seçenekler **options** isimli bir dizi oluşturur.İlk seçenek **options[0]** şeklinde belirtilir. **options[0].text** ise, ilk seçeneği metinsel değer olarak ifade eder. Önceki uygulamada kullanıcının seçtiği seçeneğin sıra numarasını ifade ediyorduk.Şimdi sıra numarası yerine, seçeneği doğrudan ifeda eden bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>Form Nesneleri</title>  <script language="Javascript1.2">  function secim(form)  {  var i=document.secenek.bolum.selectedIndex  var y=document.secenek.bolum.options[i].text  alert(y)  }  </script>  </head>  <body>  <form name="secenek">  <select name="bolum">  <option>Bilgisayar  <option>Elektronik  <option>Otomativ  <option>Makina  <option>Insaat  </select>  <input type="button" value="Hangi seçenegi seçtigimi göster" onClick="secim()">  </form>  </body>  </html> |

Kullanıcının seçtiği seçenek sıra sayısı cinsinden, **selectedIndex** ile ifade edildi.Sonra seçilen sıra numarası **options** dizini ve **text** komutu sayesinde metinsel değere dönüştürüldü.Eğer **text** yerine **index** komutu kullansaydık, yine sıra numarası elde edilirdi.Bunu örnekte deneyebilirsiniz.

Ders 20: JS Nesneleri

Şimdiye kadar ele aldığımız bütün nesneler, javascript’e html ve browser tarafından sunuluyordu.Fakat javascript, kendi kendisine nesneler de oluşturabilir.Örneğin, javascript ile dizi değişkenleri oluşturuldukları anda bunlar Array nesnesi olurlar, yeni özellikleri ve metodları olur.

Bu aşamada string nesnesini ele alalım.Javascript bir alfanümerik değişkeni ele aldığı anda bu onun için bir string nesnesi olur ve dolayısıyla, bir değişkenden öte bazı özellikler ve metodlar kazanır.

Daha önceki derslerimizde string değişkenler tanımlanmıştı.Örneğin;var isim = "Mehmet", içeriği "Mehmet" olan string(karakter) bir değişkendir.Aynı zamanda strind değişkenler, aşağıdaki şekilde de tanımlanabilir.

var isim = new String("Mehmet")

**"new"** javascript‘e yeni bir nesne oluşturması için verilen komuttur.Burada javascript’e yeni bir string nesnesi oluşturmasını, bunun içeriğini "Mehmet" yapmasını, ve bu nesneyi, "isim" isimli değişkene atamasını söylüyoruz.Değişken oluşturma açısından ikisi arasında hiç bir fark olmamakla birlikte, bu ikinci yöntem bize, alfanümerik değişkenlerin özel niteliğini, nesne olduklarını gösteriyor.Bu sayede alfanümerik değişkenlerin yani String nesnelerinin bazı ilave özelliklerinden yararlanabiliyoruz.Bu özelliklerin sık kullanılanlarını şöyle sıralayabiliriz:

|  |  |
| --- | --- |
| length | Nesnenin uzunluğunu belirtir. Örneğin isim.length, bize 6 değerini verir. |
| charAt(i) | i ile belirttiğiniz pozisyondaki karakteri verir. isim.charAt(1) bize "e" değerini verir. |
| indexOf(nnn) | nnn ile belirttiğiniz karakterlerin string içinde ilk geçtiği konumun endeksini verir. isim.indexOf("e") ifadesi, 1 değerini verir. |
| lastIndexOf(nn) | nnn ile belirttiğiniz karakterlerin String içinde geçtiği son konumun endeksini verir.isim.lastIndexOf("e"), bize 4 değerini verir. |
| substring(i,j) | String nesnesinin i’de başlayan ve j’de biten bölümünü verir.isim.subtring(0,3) ifadesi bize "Meh" değerini verir. |

String nesnesinin kendine özgü metodları vardır.Bunlar:

|  |  |
| --- | --- |
| .bold() | Bağladığınız String nesnesini koyu yapar. Örneğin isim.bold(), size "<b>Mehmet</b>" metnini verir. |
| .fontcolor("renk") | String nesnesinin görüntülenme rengini belirler. Örneğin isim.fontcolor("red") size "<font color ="red">Mehmet</font>" değerini verir. |
| .fontsize("ölçü") | String nesnesinin görüntülenmesinde harf büyüklüğünü belirler. Örneğin isim.fontsize("24") size "<font size="24">Mehmet</font>" değerini verir. |
| .italics() | String nesnesinin itelik harfle görüntülenmesini sağlar. Örneğin isim.italics() size "<i>Mehmet</i>" değerini verir. |
| .toLowerCase() | String nesnesinin görüntülenmesi sırasında bütün harflerinin küçük harf olmasını sağlar. Örneğin isim.toLowerCase () size "mehmet" değerini verir. |
| .toUpperCase() | String nesnesinin görüntülenmesi sırasında bütün harflerinin büyük harf olmasını sağlar. Örneğin isim.toLowerCase () size "MEHMET " değerini verir. |

Yapacağınız uygulamalarda bunları kullanarak değişik etkiler yaratabilirsiniz.Bu özellikler genelde formdan bilgi girişlerinde bir standart sağlamak için kullanılır.Ziyaretçi nasıl giriş yaparsa yapsın, girilen bilgiler sizin istediğiniz düzene sokulabilir.

Ders 21: DHTML

Aslında DHTML diye bir dil veya standart yoktur.DHTML, html e ek etiket getirmez; mevcut etiketlere ilave özellik ve nitelik kazandırır.Bunu CSS (Cascading Style Sheets) teknolojisi sağlar.Ayrıca DHTML yoluyla, html in bilinen etiketlerinin bilinen özellikleri, ziyaretçinin sayfanızda yapacağı tercihlere, tutum ve davranışlarına göre dinamik olarak değişmesini sağlayabilirsiniz.Başka bir deyişle, DHTML, html etiketlerinin program yoluyla dinamik hale getirilmesi demektir.

Hareketli web sayfaları oluşturmak için html, CSS ve javascript kodlarını bir şekilde ilişkilendirmek gerekir.Burada DOM(Document Object Model-Döküman Nesne Modeli) devreye girer. DOM bir modelleme şeklidir, sayfa nesnelerinin belli bir hiyerarşi içinde tanımlanmasıdır. Sayfa nesnelerini komutlandırmak için, önce tanımlamamız gerekir. Bu tanımlama için, üzerinde işlem yapılacak her html nesnesi id="..." parametresiyle isimlendirilir.Bu isim kullanılarak nesnenin parametreleri, CSS özellikleri veya metinsel değerleri üzerinde değişiklik yapılır.

Html nesnesinin parametresi üzerinde oynanan bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>DHTML-Parametre Degisimi</title>  <script language="Javascript1.2">  function degis()  {  resim.src="../resimler/pencilman2.gif"  }  function geridon()  {  resim.src="../resimler/pencilman.gif"  }  </script>  </head>  <body>  Resmin degismesi için üzerine gelin...  <br><br><br><br>  <img id="resim" onMouseOver="degis()" onMouseOut="geridon()" src="../resimler/pencilman.gif" border="0">  </body>  </html> |

CSS özellikleriyle oynamak için kullanılan yapı :**nesnenin ismi.style.css özelliği**=değer şeklindedir.Şimi bunu daha iyi anlatmak için bir örnek yapalım.

|  |
| --- |
| <html>  <head>  <title>DHTML-CSS Özellikleri</title>  <script language="Javascript1.2">  function buyut()  {  resim.style.width="150" }  function kucult()  {  resim.style.width="50" }  function degis()  {  resim.src="../resimler/pencilman2.gif" }  function son()  {  resim.src="../resimler/pencilman.gif" }  </script>  </head>  <body>  <img id="resim" onMouseOver="degis()" onMouseOut="son()" onClick="buyut()" ondblclick="kucult()" src="../resimler/pencilman.gif" border="0">  <br><br>Resmin degismesi için üzerine gelin.  <br>Resmin büyümesi için üzerine tiklayin.  <br>Resmin küçülmesi üzerine çift tiklayin.  </body>  </html> |

Örnekte CSS derslerinde gördüğümüz width özelliği kullanıldı.Sizde benzer örnekler yaparak konuyu daha iyi anlayabilirsiniz.

Dersimizin başında söylediğimiz gibi aslında DHTML, html, CSS ve javascript'in birlikte kullanılması sonucu dinamik sayfalar yaratmaktır.Verdiğimiz örneklerde javascript kodları ile hem html hem de CSS özellikleri üzerinde değişiklikler yapıldı.Böylece javascript dersleri tamamlanmış oldu.