JavaScript

Hafta 4

JavaScript nedir?

- Javascript, HTML'in bir parçasıdır ve içinde bulunduğu HTML ile birlikte Web Browser programı tarafından yorumlanır.
- Javascript programı yazmak demek, bir Web sayfası hazırlamak ve bu sayfadaki HTML kodlarının arasına Javascript kodları gömmek demektir.
- İlk JavaScript içeren sayfamız, burada JavaScript kodları HTML ile iç içer girmiş konumdadır.
  - html örnekler\örnek30.html

<SCRIPT>

- Kural olarak JavaScript kodları, HTML'in içinde <SCRIPT>…</SCRIPT> etiketlerinin arasına gömülür. örnek31.html
- Genellikle <SCRIPT> kodları <HEAD> içerisine konulur.
- JavaScript kodları başka dosyada yazılıp HTML içerisine
  `<SCRIPT SRC="merhaba.js" LANGUAGE="JavaScript">`
  şeklinde de eklenebilir. örnek32.html

Yorum Satırları – Eski Browser

- `<!-- Javascript kodunu eski sürüm Browserlardan gizleyelim`

```
function merhaba(){
   alert ("Merhaba Dünya!")
}
// kod gizlemenin sonu -->
```

- Bu bölümün başında ve sonunda yer alan "<!--" ve "-->" işaretlerinin arasındaki her şey, eski sürüm Browserlar tarafından HTML dili kurallarına göre "yorum" sayılacak ve görmezden gelinecektir

Yorum satırları

- Javascript dilinin yorumları ise "//" işaretiyle başlar ve satır sonunda sona erer.
- Eğer yorumlarınız bir satırdan uzun olacaksa, bunu şöyle belirtebilirsiniz:

```
/*  yorumun birinci satırı
    yorumun ikinci satırı
    yorumun üçüncü satırı */
```


JavaScript Yüklenmesi

- JavaScript kodları, ziyaretçinin sayfanızda bir yeri tıklaması veya klavyede bir tuşa basmasıyla harekete geçer veya
- HTML sayfası ziyaretçinin Browser'ında görüntülendiği anda otomatik olarak çalışmaya başlar.
- Otomatik çalışan JavaScript kodu ise iki ayrı yöntemle çalıştırılabilir:
  - HTML kodları icra edilmeden önce (yani sayfanız ziyaretçinin Web Browser'ında görüntülenmeden önce

    örnek33a.html
  - Sayfa görüntülendikten sonra. örnek33b.html

JavaScript'in Temel İlkeleri

- JavaScript dilinin değişkenleri, metodları ve nesnelerini belirleyen isimlere Belirleyiciler (**Identifier**) denir.
- Bu sınıfa giren bütün kelimeler ya harfle ya da alt çizgi (_) ile başlar. Rakam veya diğer işaretler birinci karakter olarak kullanılamaz, fakat daha sonra kullanılabilir.
- JavaScript, aynı kelimenin büyük harfle yazılanı ile küçük harfle yazılanını farklı isimler olarak görür.
- Bu sınıfta giren kelimelerin içinde boşluk olamaz.
- JavaScript kodlarınız sizin bilgisayarınızda değil, ziyaretçinin bilgisayarında çalıştırılacağına göre, kullandığınız karakterlerin ziyaretçinin bilgisayarında nasıl bir değer taşıyacağını düşünmeniz gerekir.
- Bu bakımdan güvenli yol, Bu sınıfa giren kelimelerde, İngilizce alfabede bulunmayan, Türkçe ve diğer dillerdeki high-ASCII karakterleri (ı, İ, ğ, Ğ, Ş, Ş ile ü, Ü, ö, Ö, ç ve Ç) kullanmamaktır.

7

Anahtar Kelimeler (Keyword)

- JavaScript dilinin önceden tanımlanmış ve programın yorumunda özel anlam kazandırılmış kelimelerine Anahtar Kelime denilir.

break	do	if	switch	typeof
case	else	in	this	var
catch	false	instanceof	throw	void
continue	finally	new	true	while
default	for	null	try	with
delete	function	return		

8

Ayrılmış Kelimeler (Reserved)

İkinci gruba girsin-girmesin bazı kelimeler, ilerde JavaScript programlama ve yorumlama işlerinde kullanılabileceği düşüncesi ile, bir kenara ayrılmıştır; JavaScript kodlarında kullanılamazlar.

abstract (soyut)
boolean (Boolean Mantığı)
break (kes)
byte (bayt)
case (hal)
catch (yakala)
char (karakter)
class (sınıf)
const (sabit)
continue (devam)
default (varsayılan)
delete (sil)
do (yap)
double (çift)
else (başka bir durum)
extends (uzanır)
false (yanlış)
final (sonuncu)
finally (sonunda)
float (kesirli)
for (için)
function (işlev)
goto (--ya git)
if (eğer)
implements (uygular)
import (ithal et)
in (içinde)
instanceof (--nın oluşumu)
int (integer, tam sayı)
interface (arayüz)
labed (etiketli)
long (uzun)
native (kendinden olan)
new (yeni)
null (boş değer)
package (paket)
private (özel)
protected (korunmuş)
public (genel)
return (dön)
short (kısa)
static (sabit)
super (kuvvet)
switch (değiştir)
synchronized (uyumlu)
this (bu)
throw (içine kat)
throws (içine katar)
transient (geçici)
true (doğru)
try (dene)
typeof (türü)
var (değişken)
void (geçersiz)
while (iken)
with (ile)

9

Değerler (Literal):

- Javascript kodu icra edildiği sırada değişmeyen rakam veya metinlere Değer denir. Javascript kodlarında beş tür değer bulunur:
  1. Tamsayı Değerler (Integer Literal):
  2. Kayan Noktalı Değerler (Floating-point literal):
  3. Boolean Mantık İfadeleri (Boolean Literal):
     - True (Doğru) değerini 1,
     - False (Yanlış) değerini 0 rakamıyla tutar.
  4. Alfanümerik (Karakter) Değerler (String literal):
  5. Özel Karakterler

10

Özel Karakterler

- `\b` - Klavyede Geri (backspace) tuşunun görevini yaptırır.
- `\f` - Yazıcıya sayfayı bitirmeden çıkarttırır (formfeed).
- `\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 almayacaktır.

11

Javascript Değişkenleri (Variable)

- Değişken, adı üstünde, Javascript yorumlayıcısı tarafından bilgisayarın belleğinde tutulan ve içerdiği değer programın akışına göre değişen bir unsurdur.
- Değişkenlerin bir adı olur, bir de değeri. Program boyunca değişkenin adı değişmez; fakat içeriği değişebilir.
- Değişkenlere isim verirken Belirleyici isimleri kurallarına riayet etmeniz gerekir.
- JavaScript, büyük harf-küçük harf ayırt ettiği (case-sensitive olduğu) için, örneğin SONUC ve sonuc kelimeleri iki ayrı değişken gösterir.

12

Değişkenler

- Değişken tanımlamak, bilgisayar programcılarına daima gereksiz bir yük gibi görünür.
- Birazdan göreceğiz, Javascript sadece değişkenleri tanımlamayı zorunlu kılmakla kalmaz, fakat nerede tanimlandığına da özel bir önem verir.
- Javascript'e bir Belirleyici'nin değişken olarak kullanılacağını bildirmek için `var` anahtar-kelimesini kullanırsınız:

```
var sonuc
var adi, soyadi, adres, siraNo
var i, j, k
var mouseNerede, kutuBos, kutuDolu
```

- Bazı diğer programlardan farklı olarak Javascript, size değişkenleri hem beyan, hem de içeriğini belirleme işini aynı anda yapma imkanı veriyor (initialization):

```
var sonuc = "Merhaba Dünya!";
var adi = "Abdullah", soyadi = "Aksak";
var i = 100, j = 0.01, k = 135;
var kutuBos = false, kutuDolu = true;
```

13

Değişkenler

- Bir değişkeni tanımlayarak içini doldurmadan (initialization'dan) önce içindeki değeri sorgulamaya kalkarsanız, Browser'ın Javascript yorumlayıcısı o noktada durur ve tanımlanmamış (undefined) değişken hatası verir.
- Javascript programlarında beş tür değişken bulunabilir:
  1. Sayı (number): -14, 78, 87678
  2. Boolean değişken: true, false
  3. Alfanümerik (String) Değişken: "Merhaba Dünya!"
  4. İşlev (Function) Değişken:
  5. Nesne (Object) değişkenleri: window, document

14

Değişkenleri İsimlendirmek

- JavaScript, değişkenlerinizi isimlendirmede rakamla ve işaretle başlamamak dışında kural koymuyor; ama iyi programcılık tekniği, değişkenlere anlaşılır ve kullanıldığı yeni belli eden isimler vermektir.
- Örneğin, **`adi`** şeklindeki bir değişken adı, çok değişkenli bir Javascript programında yanıltıcı olur. **`musteri_adi`** veya **`musteriAdi`** çok daha uygun olabilir.
- Ayrıca değeri **`Evet`** (=doğru, true) veya **`Hayır`** (=yanlış, false) olan Boolean değişkenlerin adlandırılmasında, örneğin, **`doluMu`**, **`tamamMi`**, **`acikMi`**, **`bittiMi`** gibi cevabı çağrıştıran değişken adları kullanabilirsiniz.

15

Yetki Alanı (Scope)

- HTML dosyasının baş tarafında, <HEAD> etiketi içinde, bir değişkeni tanımladığınızı ve ona bir değer verdiğinizi düşünün.
- Daha sonra yazacağınız bütün fonksiyonlarda veya değerini belirleyebileceğiniz otomatik fonksiyonlarda (metod'larda) bu değişkeni bir daha tanımlamaya ve değerini belirlemeye gerek kalmadan kullanabilirsiniz; çünkü Javascript açısından bu değişken genel (global) değişken sayılır. örnek34a.html
- HTML'in gövde kısmında (<BODY> etiketi içinde) bir fonksiyon yazdığımızı ve bu fonksiyonun içinde bir değişken tanımladığımızı düşünün.
- Daha sonra yazacağınız bir fonksiyonda bu değişkeni kullanamazsınız; çünkü JavaScript bir fonksiyonun içindeki değişkeni yerel (local) değişken sayar ve kendi yetki alanı (scope'u) dışında kullanmanıza izin vermez. örnek34b.html

16

Yetki Alanı - Sonuç

- Bu alıştırmanın öğrettiği kural şöyle özetlenebilir: Bir genel değişken, bir fonksiyon içinde yerel olarak değiştirilebilir; ama onun genel değeri diğer fonksiyonlar için geçerli kalır.
- Javascript programlarınızı yazdığınız zaman genel değişkenleriniz beklediğiniz değeri vermiyorsa bu değeri bir fonksiyonun yerel olarak, sırf kendisi için, değiştirip değiştirmediğine bakmalısınız.
- Bu tür hatalardan kaçınmanın bir yolu, yerel değişkenlerinizle genel değişkenlerinize farklı isimler vermektir.

17

Dizi Değişkenler

- **Dizi değişkenler ise birden fazla değer tutabilirler.** Bir dizi değişkenin bireylerinin değerlerini dizi-değişkenin indeksi ile çağırabilir veya belirleyebilirsiniz.

```
var dizinin_Adı = new Array(unsur1, unsur2, unsur3...unsurN)
var kusDizi = new Array("bülbül", "kanarya", "muhabbet",
                        "papagan");
```

- Diziler sıfırdan itibaren numaralanır. Örneğimizde dizinin
  - birinci üyesi `kusDizi[0]`,
  - ikinci üyesi `kusDizi[1]`,
  - üçüncü üyesi `kusDizi[2]` diye anılır.
- Burada örneğin `kusDizi[2]` değişkeni "muhabbet" değerini taşır.
- Dizi değişkeni eleman sayısını vererek de deklere edebilirsiniz

```
var weekday=new Array(7);
```

html örnekler\örnek34c.html

18

Dizi değişken Metotları

- **join( )** : Dizi değişkenin elemanlarını birleştirmek için kullanılır, sonuç String olur.

```
var a = [1, 2, 3]; // Create a new array with these three elements
var s = a.join( ); // s == "1,2,3"
```

html örnekler\örnek34e.html

- **reverse( )** : Dizi değişkenleri elemanlarının sırasını tersine çevirmek için kullanılır.

```
var a = new Array(1,2,3);   // a[0] = 1, a[1] = 2, a[2] = 3
a.reverse( );               // now a[0] = 3, a[1] = 2, a[2] = 1
var s = a.join( );          // s == "3,2,1"
```

html örnekler\örnek34g.html

Dizi değişken Metotları

- **split( )** : Bir stringi bazı parçalayıcı karakterler kullanarak (genellikle boşluk, virgül, noktalı virgül gibi) dizi değişkene parçalar. **join()** in tam tersidir.

```
var line1 = "a b c d e f"
var ar1 = new Array()
ar1 = line1.split(" ")  // ar1[0]="a", ar1[1]="b", ...
```

- **sort( )** : Dizi değişkenin elemanlarını sıralamak için kullanılır.

```
var a = new Array("banana", "cherry", "apple");
a.sort( );
var s = a.join(", ");          // s == "apple, banana, cherry"
```

html örnekler\örnek34f.html

Dizi değişken Metotları

- **concat( )** : Dizi değişkene eleman eklemek için kullanılır.

```
var a = [1,2,3];
a.concat(4, 5)          // Returns [1,2,3,4,5]
a.concat([4,5]);        // Returns [1,2,3,4,5]
a.concat([4,5],[6,7])   // Returns [1,2,3,4,5,6,7]
a.concat(4, [5,[6,7]])  // Returns [1,2,3,4,5,[6,7]]
```

html örnekler\örnek34d.html

- **slice(a,k)** : Dizi değişkenin a. Elemanların itibaren k eleman fakat k. dahil olmayacak şekildeki elemanlarını alarak yeni bir dizi değişken oluşturulur. -1 son elemanı ifade ederken, -3 sondan üçüncü elemanı ifade eder.

```
var a = [1,2,3,4,5];
a.slice(0,3);    // Returns [1,2,3]
a.slice(3);      // Returns [4,5]
a.slice(1,-1);   // Returns [2,3,4]
a.slice(-3,-2);  // Returns [3]
```

html örnekler\örnek34h.html

Dizi değişken Metotları

- **splice(m,n)** : dizi değişkenin **m.** elemanından sonraki **n** tane elemanını siler. Tek argüman alırsa **m.** Elemandan sonraki elemanları siler.

```
var a = [1,2,3,4,5,6,7,8];
a.splice(4);     // Returns [5,6,7,8]; a is [1,2,3,4]
a.splice(1,2);   // Returns [2,3]; a is [1,4]
a.splice(1,1);   // Returns [4]; a is [1]
```

html örnekler\örnek34k.html

- Bu metot aynı zamanda eleman eklemek içinde kullanılır:

```
var a = [1,2,3,4,5];
a.splice(2,0,'a','b');        // Returns []; a is [1,2,'a','b',3,4,5]
a.splice(2,2,[1,2],3);        // Returns ['a','b']; a is [1,2,[1,2],3,3,4,5]
```


Dizi değişken Metotları

- **push()** and **pop()** dizi değişkenler yığıt (stack) mış gibi işlem yaptırır.
- **push()** dizi değişkenin sonuna eleman ekler, **pop()** ise dizi değişkenin sonundan bir eleman geri getirir.

html örnekler\örnek34l.html html örnekler\örnek34m.html

```
var stack = [];       // stack: []
stack.push(1,2);      // stack: [1,2]     Returns 2
stack.pop( );         // stack: [1]       Returns 2
stack.push(3);        // stack: [1,3]     Returns 2
stack.pop( );         // stack: [1]       Returns 3
stack.push([4,5]);    // stack: [1,[4,5]] Returns 2
stack.pop( )          // stack: [1]       Returns [4,5]
stack.pop( );         // stack: []        Returns 1
```


Dizi değişken Metotları

- **shift( )** and **unshift( )** metotları **push()** ve **pop()** gibi davranır, fakat bu işlemleri dizinin başlangıç tarafında gerçekleştirir. **shift()** başlangıçtaki elemanı geri dönderir, **unshift()** ise baş tarafa eleman ekler.
- **unshift( )** sadece Netscape tarayıcısında çalışır.

html örnekler\örnek34n.html html örnekler\örnek34o.html

```
var a = [];             // a:[]
a.unshift(1);           // a:[1]          Returns: 1
a.unshift(22);          // a:[22,1]       Returns: 2
a.shift( );             // a:[1]          Returns: 22
a.unshift(3,[4,5]);     // a:[3,[4,5],1] Returns: 3
a.shift( );             // a:[[4,5],1]    Returns: 3
a.shift( );             // a:[1]          Returns: [4,5]
a.shift( );             // a:[]           Returns: 1
```

Dizi değişken Metotları

- `toString( )` and `toLocaleString( )` metotları dizi değişkenleri `String`'e dönüştürür.

```
[1,2,3].toString( )          // Yields '1,2,3'
["a", "b", "c"].toString( )  // Yields 'a,b,c'
[1, [2,'c']].toString( )     // Yields '1,2,c'
```

25

Çok Boyutlu dizi değişkenler

- Çok boyutlu dizi değişkenler; dizi değişkenlerin dizi değişkenleri olarak tanımlanır.

```
Brady = new Array(3)
for (i = 0; i < Brady.length; ++ i)
Brady [i] = new Array(3);

Brady [0] [0] = "Marsha";
Brady [0] [1] = "Carol";
Brady [0] [2] = "Greg";
```

- Bunun yanında iç içe geçmiş şekilde dizi değişkenler tanımlanabilir.

```
var matrix = [[1,2,3], [4,5,6], [7,8,9]];
matrix[0][2]=3
```

html örnekler\örnek34p.html

26

Atama (Assignment) İşlemi

- Tahmin ettiğiniz gibi bunu eşittir (=) işaretiyle yaparız:

```
var adi = "Ahmet";  soyadi = "Arif";
var siraNo = 123; sigortaNo = "A123-2345-234";
```

- Javascript'te bir değişkene değer atarken, bu değeri mevcut bir veya daha fazla değişkenden de alabilirsiniz:

```
var i = j + k
var indexNo = siraNo + kategoriNo
var tutariTL = birimFiyatTL * adet
```

27

Aritmetik İşlemleri:

- Javascript, dört temel işlemi yapabilir. toplama işlemini artı işaretiyle (+), çıkartma işlemini eksi işaretiyle (-), çarpma işlemini yıldız (asterisk, *) işaretiyle, ve bölme işlemini düz bölü işaretiyle (/), % ile kalan bulma işlemi yaptırırsınız.
- Arttırma ve azaltma işlemleri, değişkenin değerini işlemde kullanmadan veya kullandıktan sonra arttırma veya azaltma işlemi yapar.
- Sonradan artırma x++, önceden artırma ++x. Sonradan azaltma x--, önceden azaltma --x.
- Toplama ve çıkartma işlemlerinde yapabileceğiniz başka bir kısaltma ise şöyle yazılır:
  x = x + y işlemini kısaltmak için x += y
  x = x - y işlemini kısaltmak için x -= y
- Javascript, alfanümerik değerlerle çıkartma, çarpma ve bölme işlemleri yapmaz; sonucun yerine NaN (Not a Number, Sayı Değil) yazar.

28

İşlemlerde Sıra:

- Aynı önceliğe sahip operatörler soldan sağa doğru işleme dahil olurlar.
- İşlem sırası için tablo ya bakınız.
- İşlem sırasında parantez varsa öncelikle parantez içerisindeki işlemler yapılır.
- İşlem sırası karmaşası yaşamamak için işlemlerin uygun parantezler ile yeniden yazılması tavsiye edilir.

```
x = a * b + c
x = a * (b + c)
x=a*b/c
x=a/c*b
```

29

Özel Sayısal Değerler

- JavaScript bazı matematikte kullanılan özel değerlere karşılık gelecek şekilde sabitler tanımlamaktadır.

Sabit	Anlamı
Infinity	Special value to represent infinity
NaN	Special not-a-number value
Number.MAX_VALUE	Largest representable number
Number.MIN_VALUE	Smallest (closest to zero) representable number
Number.NaN	Special not-a-number value
Number.POSITIVE_INFINITY	Special value to represent infinity
Number.NEGATIVE_INFINITY	Special value to represent negative infinity

30

Karşılaştırma İşlemleri:

- Javascript'in karşılaştırma operatörleri genellikle "if" (eğer..ise) ifadesiyle birlikte kullanılır; ve bu soruyu soran satıra "true" (doğru) veya "false" (yanlış) sonucunu verir. Önce, bu işlemleri yaptırtan operatörleri ve işlevlerini sıralayalım:
- == Eşit operatörü. İşaretin sağında ve solundaki değerlerin eşit olması halinde true (doğru) sonucunu gönderir.
- != Eşit değil operatörü. İşaretin sağında ve solundaki değerlerin eşit olmaması halinde true (doğru) sonucunu gönderir.
- > Büyüktür operatörü. Soldaki değer, dağdaki değerden büyük ise true (doğru) sonucunu gönderir.
- < Küçüktür operatörü. Soldaki değer, dağdaki değerden küçük ise true (doğru) sonucunu gönderir.
- >= Büyük veya eşit operatörü. Soldaki değer, dağdaki değerden büyük veya bu değere eşit ise true (doğru) sonucunu gönderir.
- <= Küçük veya eşit operatörü. Soldaki değer, dağdaki değerden küçük veya eşit ise true (doğru) sonucunu gönderir.

31

Alfanümerik (String) İşlemler

- Javascript'in alfanümerik değişkenlerin değerleri ile sadece toplama işlemi yaptığını söylemiştik.
- Bu durumda buna toplama değil birleştirme, ekleme işlemi denir.
  - html örnekler\örnek36.html
- Dikkat : Bu arada bir değişkene string değer atandığı zaman o değişkenin otomatik olarak bir karakterler dizisi olarak algılanmayacağı da önemli bir özelliktir. Yani;

```
ad="Serdar";
```

şeklinde bir tanımlamada `ad[0]=S , ad[1]=e ...` geçerli olmayacaktır.

32

String

- Bir stringin içinden bir harfi ya da harf grubunu almak istediğimizde kullanmamız gereken komut `substring` komutudur.

```
harf=ad.substring(m,n);
```

Buradaki m, başlangıç karakterinin, n ise bitiş karakterinin indeks değerini gösterir. Yani

```
var ad ="Serdar";
harfler=ad.substring(0,2);
```

deseydik harfler değişkeni `"Ser"` değerini içerecekti.

- Bu işlemin tam tersi de mümkündür. Yani girilen bir karakterin kaçıncı karakter olduğunu bulmak. Bunun için de `indexOf` komutunu kullanıyoruz.

```
sayi=ad.indexOf("e");
```

yazdığımızda `sayi` değişkenine 1 değeri atanır.

- Ayrıca bu değişkenin uzunluğunu bulmak için `length` komutu kullanılır.

```
sayi=ad.length;
```

dediğimizde `sayi` değişkeninde 6 değeri bulunacaktır.

- html örnekler\örnek36a.html

33

Şartlı İşlemler

- Javascript'te karşılaştırma yaparken şartlı (..ise ..yap!) işlemler de yaptırabilirsiniz. Şartlı işlemlerde ? (soru işareti) ve : (iki nokta üst üste) işaretlerini kullanırsınız.

```
sonucMsg = (y==x) ? "y degiskeni x degiskenine esit !" : "y degiskeni x degiskenine esit degil !";
alert(sonucMsg);
```

34

Mantıksal İşlemler

- Javascript, karşılaştırma işlemini Boolean (.. ve .. doğru ise ... yap!) mantıksal işlemlerini kullanarak da yapabilir.
- Javascript'in Boolean mantığını sorgulama işaretleri şunlardır:
- && Mantıksal Ve: iki koşul da doğru.
- || Mantıksal Veya: ya birinci, ya da ikinci koşul doğru.
- ! Mantıksal Değil: Tek koşula uygulanır; koşul doğru ise sonuç false (yanlış), koşul yanlış ise sonuç true (doğru) olur.
- Bu mantığı, şimdi Boolean ifadesi haline getirelim:

```
A>D && D>E
E<=F ||  x==y
```

- Birinci ifadenin değili

```
!(A>D) || !(D>E)
```

olur

35

Program Akış Denetimi

- Program içerisinde belli bir şart sağlandığında programın bir kısmının çalıştırılması istenebilir.
- Bu durumda if deyiminden yararlanılır.

```
if (şart) {
   çalıştırılması istenen ifade;
}
```

- html örnekler\örnek37.html

36

if … else …

- İstenilen şart sağlandığında bir program parçasının sağlanmadığında ise başka bir program parçasının çalışmasını istiyorsanız:

```
if (şart) {
   Çalıştırılması istenen ifade;}
else{
  Çalıştırılması istenen başka ifade;
}
```

37

Döngü Denetimi

- if yapısında çalışması istenilen kısım bir kere çalışır. Bazı durumlarda, istenilen program parçasının birden fazla çalıştırılması istenir.
- Bu işlem `for( I; II; III)` ile yapılır.
  - I: Döngü değişkeni ve başlangıç değeri
  - II: Sonlandırma koşulu
  - III: Bir sonraki aşamaya geçişteki artma miktarı

```
for(i=1; i< 11; i++){
     document.writeln(i+". iterasyon");
}
```

- html örnekler\örnek39.html
- html örnekler\örnek39a.html
- html örnekler\örnek39b.html

38

While döngüsü

- for döngüsünde sayaç değişkeninin belli değerleri için istenilen program parçasının çalıştırılması istenmektedir.
- Bazı durumlarda bir koşul sağlandığı sürece istenilen program parçasının çalışması istenir.
- Bunun için

```
while (şart) {
  Çalıştırılması istenen ifade;
  Sayaç değişkeni ile şart ifadesi hesaplanmalıdır;
}
```

- html örnekler\örnek40.html
- html örnekler\örnek40a.html

39

for (x in mycars)

- Dizi değişkenin içerisindeki her bir eleman için döngü kurar. Burada x dizi değişkenin index değeridir.

```
var cars = new Array();
cars['Sedan']= "Volkswagen Passat";
cars['Hatchback'] = "Toyota Auris";
cars['Spor'] = "BMW M6";

for (var x in cars){
   document.write(cars[x] + "<br />")
}
```

- html örnekler\örnek40b.html

40

do … while (şart)

- while döngüsünün başlayabilmesi için öncelikle şartın ilk anda sağlanması gerekmektedir.
- Eğer sağlanmazsa döngü hiç başlamaz.
- Bu durumda, döngünün en az bir defa çalışması ve sonrasında sağlanması istenen şartın kontrol edilerek döngüye devam edilmesi istenebilir.
- Bu ise

```
do{
      Çalıştırılması istenen ifade;
      Sayaç değişkeni ile şart ifadesi hesaplanmalıdır;
}
while (şart)
```

- html örnekler\örnek41.html
- html örnekler\örnek41a.html

41

break, continue

- Şartlı döngüde, tekrar eden iş, şartın yerine geldiği noktada kendiliğinden kesilecektir.
- Aynı otomasyonu `for` döngüsünde `break` (kes) ve `continue` (devam et) komutlarıyla biz de sağlayabiliriz.
- JavaScript, `break` ile karşılaştığı anda döngüyü keser ve icraata döngüden sonra gelen ilk ifadeden devam eder.
- `continue` ise JavaScript'in döngünün o andaki adımını durdurup, döngünün başına dönmesini sağlar; döngü sayacın bir sonraki değeriyle işleme devam eder.

- html örnekler\örnek42.html
- html örnekler\örnek42a.html
- html örnekler\örnek42b.html

42

switch

- Javascript'in `switch (değişken)` komutu, programın bir değişkenin değerine göre, belirli bir durum dahilinde bir işin yapmasını sağlar.

```
switch (değişken){
      case "1": ifade; break;
      case  2 : ifade; break;
}
```

- html örnekler\örnek43.html

43

JavaScript'te Fonksiyon

- Şu ana kadar gördüğünüz işlerin çoğu bir kere başvurulan işlerdi; fakat çoğu zaman sayfanızdaki bir JavaScript işleminin defalarca yapılması gerekebilir.
- **Defalarca yapılması istenen işlemler bir grup haline getirilerek fonksiyon adını verdiğimiz bir grup oluşturulur.**
- Genel hatlarıyla fonksiyon, şu formüle göre yazılır:

```
function fonksiyon_adı(arg1,arg2,...,argN){
      işlemler;
      return sonuç;
}
```

- html örnekler\örnek44.html
- html örnekler\örnek44a.html
- html örnekler\örnek44b.html

44

charAt(i), parseInt(S,n), parseFloat(S)

- `charAt(i)` : Stringindeki i. pozisyondaki Karakterini geri gönderen fonksiyondur.

```
var s="Merhaba";
document.write(s.charAt(1))  // Ekrana e yazdırır.
```

- Alfanümerik değerleri tam sayıya çevirmek için `parseInt(String,n)` ifadesini kullanırız. Buradaki n, sayının hangi tabanda yazılması gerektiğini ifade eder (8, 10, 16 lık sistemlerde).

```
var a="18.4";
document.write(parseInt(a,10)); // 18
```

- Alfanümerik değerleri ondalı sayıya (floating point) çevirmek için `parseFloat(String)` ifadesini kullanırız.

45

Saat ve Tarih

- Browser'ın işletim sisteminden, işletim sisteminin de bilgisayarın temel girdi/çıktı işlemlerini yapan BIOS'tan saati ve tarihi içeren bilgiyi almasını sağlar.
- Buradaki metod `Date` (günün tarihi) adını taşıyor, ama `Date()` JavaScript açısından class (sınıf) sayılır ve edinilen bilgi sadece ay, gün ve yıl bilgisini değil, o andaki saati, dakikayı ve saniyeyi de içerir.

```
new Date() // current date and time
new Date(milliseconds) //milliseconds since 1970/01/01
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

- Tarih başlatıcıları için örnekler

```
today = new Date()
d1 = new Date("October 13, 1975 11:13:00")
d2 = new Date(79,5,24)
d3 = new Date(79,5,24,11,33,0)
```

- Tarihi 14 Ocak 2011 olarak ayarlamak için

```
var myDate=new Date();
myDate.setFullYear(2011,0,14);
```

- Tarihi beş gün sonrasına ayarlamak

```
var myDate=new Date();
myDate.setDate(myDate.getDate()+5);
```

46

Saat ve Tarih

- İki tarihi karşılaştırır

```
var myDate=new Date();
myDate.setFullYear(2010,0,14);
var today = new Date();
if (myDate > today)
{
  alert("Today is before 14th January 2010");
}
else
{
  alert("Today is after 14th January 2010");
}
```

- Bazı get metotları
  - `getYear` : Yıl (1900'den sonra)
  - `getMonth` : Ay (0=Ocak - 11=Aralık)
  - `getDay` : Gün (0-6)
  - `getDate` : Gün (0-6)
  - `getTime` : 1970'den bu zamana kadar geçen zamanı mili saniye olarak verir.
  - `getHours` : Saat (0-23)
  - `getMinutes` : Dakika (0-59)
  - `getSeconds` : Saniye (0-59)

örnek45.html örnek45a.html örnek45b.html örnek45c.html

Tarih bilgisini UTC (GMT) ye göre verir.

```
<script type="text/javascript">
  var d=new Date();
  document.write("Original form: ");
  document.write(d + "<br />");
  document.write("To string
        (universal time): ");
  document.write(d.toUTCString());
</script>
```

`toString()`, `toLowerCase()`, `toUpperCase()`

- `toString()` metodu: Kelime anlamı String'e çevir, alfanümerik'e çevir olan bu metotla, saat nesnesinin tamamen kendine özgü biçimi, Javascript tarafından HTML'in anlayabileceği şekle çevrilmiş olur.

```
var fruits = ["Banana", "Orange", "Apple",
  "Mango"];
document.write(fruits.toString());
```

- `toLowerCase()` (küçük harfe çevir) ve `toUpperCase()` (büyük harfe çevir) metotları sadece alfanümerik (String) değerlere uygulanabilir.

```
var txt="Hello World!";
document.write(txt.toLowerCase() + "<br />");
document.write(txt.toUpperCase());
```

48

String Nesnesi Özellikleri

```
var KitapAdi="Gazap Üzümleri";
```

- `.indexOf(nnn)` nnn ile belirttiğiniz karakterlerin String içinde ilk geçtiği konumun indeksini verir.
  `kitapAdi.indexOf("a");`
- `.lastIndexOf(nn)` nnn ile belirttiğiniz karakterlerin String içinde geçtiği son konumun indeksini verir.
  `kitapAdi.lastIndexOf("a");`
- `.bold()` Bağladığınız String nesnesini koyu yapar. Örneğin
  `kitapAdi.bold()`
  `<B>Gazap Üzümleri</B>` etkisini verir.
- `.fontcolor("renk")` String nesnesinin görüntülenme rengini belirler. Örneğin
  `kitapAdi.fontcolor("red")`
  `<FONT COLOR="red">Gazap Üzümleri</FONT>` değerini verir.
- `.fontsize("ölçü")` String nesnesinin görüntülenmesinde harf büyüklüğünü belirler. Örneğin
  `kitapAdi.fontsize("24")`
  `<FONT SIZE="24">Gazap Üzümleri</FONT>` değerini verir.
- `.italics()` String nesnesinin itelik harfle görüntülenmesini sağlar. Örneğin
  `kitapAdi.italics()`
  `<I>Gazap Üzümleri</I>` değerini verir.

49

setTimeout("fonksiyon_adı", milisaniye)

- İstenilen fonksiyonun istenilen milisaniye aralıklarla çağrılmasını sağlayan bir fonksiyondur.
  `t=setTimeout("timedCount()",1000)`
- Çağrımın sonlandırılması için
  `clearTimeout(t)`
- örnek45.html içerisinde aşağıdaki gibi düzenleme yaparak saat 6:30 da başka bir iş yapılması ayarlanabilir.

```
if ((saat.getHours() == 6) && (saat.getMinutes() == 30)){
        document.saatForm.saatkutusu.value = "Alarm !!!";
}
```

- html örnekler\örnek46.html

50

Hazır Matematik Fonksiyonları

Operatör	Anlamı	Operatör	Anlamı
Math.PI	Pi sayısı	Math.cos(x)	x'in cosinüsü
Math.E	E sayısı	Math.tan(x)	x'in tanjantı
Math.Sqrt(x)	x'in karekökü	Math.asin(x)	x'in arcsinüsü
Math.log(x)	x'in doğal logaritması	Math.acos(x)	x'in arccosinüsü
Math.pow(x,y)	x'in y inci kuvveti	Math.atan(x)	x'in arctanjantı
Math.sin(x)	x'in sinüsü	Math.abs(x)	x'in mutlak değeri

51

Hazır Matematik Fonksiyonları

Operatör	Anlamı	Operatör	Anlamı
Math.max(x,y)	x, y nin maksimumu	Math.floor(x)	x'sayısından küçük ve en yakın tamsayı
Math.min(x,y)	x, y nin minimumu	Math.random()	0" ile"1" arasında rasgele bir sayı değeri
Math.ceil(x)	x'sayısından büyük en yakın tamsayı	Math.exp(x)	x'in exponansiyeli

html örnekler\örnek46m.html

52

Rastgele sayı üretme

- `Math.random()` fonksiyonuyla 0 ile 1 arasında rastgele bir sayı üretilir.
- `Math.floor((n+1)*Math.random())` fonksiyonuyla 0 ile n arasında herhangi bir rastgele bir sayı üretilmiş olur.
- html örnekler\örnek46ma.html

53

try … catch(err) …

- Yazılan kod içerisinde muhtemel bir hata olması kaçınılmaz ise bu hatanın oluşması durumunda programın çalışmasının kesilmemesi, buna karşılık bir mesaj veya başka bir işlem yapılması isteniyorsa `try … catch …` ifadeleri kullanılır.

```
try{
…
}
catch(err){
…
}
```

- `örnek46a.html` `örnek46b.html`
- Bir hata oluşması durumunda `onerror` kullanılarak hata ayıklamak için örnek46c.html

54

throw

- `throw` ifadesi kendi istisna durumu (ender karşılaşılan durum) oluşturmanıza imkan verir.
- Bu şekilde akışı daha detaylı kontrol ederek, bir özel hata oluştuğunda hatayı kendi belirlediğiniz şekilde ortadan kaldırabilir veya kullanıcıya daha detaylı bilgi sunabilirsiniz.

```
throw(exception)
```

html örnekler\örnek46d.html

55

Nesneler, Olaylar ve Özellikleri

- Javascript programcılığında nesne (object), ve nesnenin özellikleri (properties), genellikle HTML belgesinin adı (name) ve değeri (value) olan her şeydir.
- Bir HTML unsurunun etiketinde NAME ve VALUE bölümleri varsa, bu unsur, Javascript için nesne sayılır.

  Bu tanıma göre Form, Javascript için bir nesnedir.

- Ayrıca Form nesnesinin bir öğesi olan INPUT, kendisi de ad ve değer alabildiğine göre, Javascript için bir nesne sayılır. Bu nesneye daima içinde bulunduğu nesne "dolayısıyla" atıfta bulunabilirsiniz.
- Bu tür atıflarda bulunurken, şu kurala uymanız gerekir:

```
nesneAdi.özellikAdi
```

56

Nesne ve Metotları

- Browser'ın masaüstündeki penceresinin bir özelliği gibi değerleri belirleyen otomatik işlevleri; nesnelerin değerlerini belirli bir düzen içinde arttıran veya azaltan süreçleri ve JavaScript'in hazır şablonlarından yeni bir nesne üreten işlemleri, metod adı altında toplarız.
- Her nesnenin kendine ait bir metodu olabilir; bir metod birden fazla nesne ile birlikte kullanılabilir.
- Bu gibi ifadeleri şöyle yazarız:

```
nesneAdi.metodAdi(argüman)
```

57

Nesne Oluşturmak

- Nesne birden fazla özelliği olan bir değişkendir. Bir insanın adı, soyadı, yaşı, kredi kart numarası gibi bilgileri tek bir değişken altında tutmak mümkündür. Bunun için de önce bunu oluşturan bir fonksiyon yazıp sonra istediğimiz değişkeni gerekli parametrelerle bu fonksiyon cinsinden tanımlamalıyız.

```
function insan(ad,soyad,yas,kartno){
    this.ad=ad;
    this.soyad=soyad;
    this.yas=yas;
    this.kartno=kartno;
}
kisi[1]= new insan("Serdar","Ünlü",22,12345678)
```

- Buradaki `this` anahtar kelimesi bu nesneye ait olan özellikler için(sadece fonksiyonun içinde) kullanılır.
- Daha sonra kişi isimli dizinin birinci elemanını insan cinsinden yeni bir nesne olduğunu belirtmek için `new` anahtar kelimesini kullanarak ve gerekli parametreleri vererek fonksiyonu çağırıyoruz.
- Daha sonra gerekli özelliklere erişmek için

```
kisi[1].ad , kisi[1].soyad ...
```

yazılır.

58

Özel Nesneler

- JavaScript'te bir önceki konuda anlattığımız gibi kendi tanımladığımız nesnelerin yanı sıra halihazırda var olan nesneler de vardır.
- Bizim için önemli olan şöyle sıralayabiliriz.
  - window
  - Frame, parent, self, _top...
  - location
  - history
  - document
  - form
  - text field
  - text area
  - checkbox
  - radio
  - password
  - select
  - button
  - submit
  - reset
  - link
  - anchor

59

Olaylar

- Browser programları kendiliklerinden veya GUI sonucu, öyle bazı olaylara (örneğin Mouse işaretçisinin bir nesnenin üzerine gelmesi veya bilgisayar kullanıcısının Mouse'un veya klavyenin bir düğmesini tıklaması gibi) yol açarlar ki, bu olay işletim sistemi-GUI-Browser yoluyla HTML belgesi (ve dolayısıyla Javascript) açısından önem taşıyabilir. Bunlara event (olay) denir.
- Kullanıcının Mouse'un bir düğmesini tıklaması, Click, bu olayı karşılayan ve olay yönlendiren metot (Event Handler) ise onClick (tıklandığında, tıklama halinde) adını taşır.
- Javascript'e bu olayın olması halinde icra edilmek üzere özel emirler verilebilir. Bu tür komutların yazılmasında şu yöntem izlenir:

```
event.fonksiyon_veya_metod(argüman)
```

60

Olaylar

- Event Handler'ları, bir tür adlandırılmış ama içi boş fonksiyonlar gibi düşünebilirsiniz.
- Programcı olarak bize düşen, bu olay karşısında olay yönlendiricisinin ne yapmasını istediğimi belirtmekten ibarettir.
- Event Handler'lar, daha önceden hazırladığımız bir fonksiyonu göreve çağırabilir; veya hemen oracıkta bir Javascript metodunu da devreye sokabiliriz. Mesela:
  <INPUT TYPE="text" SIZE=50 MAXLENGTH=100 NAME="soyadi" onChange="denetle(this)">
- Ziyaretçi bu INPUT kutusuna soyadını yazdığında, kutunun içeriği değişmiş olacak ve bu Change (değişiklik) olayı, kendisini yöndendiren onChange sayesinde, önceden hazırladığımız "denetle()" isimli fonksiyonu çağıracaktır.
- Burada gördüğünüz "this" (bu) kelimesi, Javascript'e fonksiyonun istediği değer kümesi olarak bu nesnenin içeriğini vermesini bildiriyor.

61

Olaylar

- `onLoad` : Sayfa yüklenmesi tamamlandığında
- `onUnload` : Sayfa yüklenmesi bittiğinde(kullanıcı sayfadan çıktığında)

- `onFocus` : Eleman focusu kazandığında
- `onBlur` : Eleman focusu kaybettiğinde

- `onChange` : Seçim yapıldığında veya metin değiştirildiğinde

- `onSubmit` : Submit(gönder) butonu basıldığında
- `onReset` : Reset düğmesi tıklandığında

- `onMouseOver` : Mouse pointer bir alan veya linkin üzerine geldiğinde
- `onMouseOut` : Mouse pointer bir alan veya linkten uzaklaştırıldığında
- `onMouseDown` : Mouse düğmesine basıldığında
- `onMouseMove` : Mouse hareket ettirildiğinde
- `onMouseUp` : Mouse düğmesi bırakıldığında

62

Olaylar

- `onClick` : Nesne üzerine mouse ile tek tıklandığında
- `onDblClick` : Nesne üzerine mouse ile çift tıklandığında

- `onAbort` : Resim yüklemesi kesildiğinde
- `onError` : Resmin veya ekranın yüklenmesinde hata oluştuğu zamanlar

- `onSelect` : Yazı seçildiğinde
- `onReSize` : window veya frame yeniden boyutlandırıldığında

- `onKeyDown` : Bir tuşa basıldığında
- `onKeyUp` : Basılı tuş bırakıldığında
- `onKeyPress` : Bir tuşa basıldığında veya basılı tutulduğunda

63

Mouse / Keyboard Özellikleri

Mouse'un sol düğmesine basılıp basılmadığını kontrol için event.button==1

Özellik	Tanım
altKey	Bir olay tetiklendiğinde ALT tuşuna basılıp basılmadığı bilgisini geri getirir.
button	Bir olay tetiklendiğinde mouse un hangi tuşuna basıldığı bilgisini geri getirir.
clientX	Bir olay tetiklendiğinde fare işaretçisinin yatay koordinat bilgisini geri getirir.
clientY	Bir olay tetiklendiğinde fare işaretçisinin dikey koordinat bilgisini geri getirir.
ctrlKey	Bir olay tetiklendiğinde CTRL tuşuna basılıp basılmadığı bilgisini geri getirir.

64

Mouse / Keyboard Özellikleri

Özellik	Tanım
metaKey	Bir olay tetiklendiğinde META tuşuna basılıp basılmadığı bilgisini geri getirir.
relatedTarget	Bir olay tetiklendiğinde olayı tetikleyen elemanı geri getirir.
screenX	Bir olay tetiklendiğinde fare işaretçisinin ekrana göre yatay koordinat bilgisini geri getirir.
screenY	Bir olay tetiklendiğinde fare işaretçisinin ekrana göre dikey koordinat bilgisini geri getirir.
shiftKey	Bir olay tetiklendiğinde SHIFT tuşuna basılıp basılmadığı bilgisini geri getirir.

65

Mouse / Keyboard Özellikleri

Özellik	Tanım
bubbles	Returns a Boolean value that indicates whether or not an event is a bubbling event. (IE de yok)
cancelable	Returns a Boolean value that indicates whether or not an event can have its default action prevented (IE de yok)
currentTarget	Returns the element whose event listeners triggered the event (IE de yok)
eventPhase	Returns which phase of the event flow is currently being evaluated (IE de yok)
target	Returns the element that triggered the event
timeStamp	Returns the time stamp, in milliseconds, from the epoch (system start or event trigger) (IE de yok)
type	Returns the name of the event

66

Örnekler

- Farenin hangi tuşuna basıldı — örnek75.html
- İlmeç koordinatları — örnek75a.html
- Basılan tuşun unicode değeri — örnek75b.html
- Ekrana göre ilmeç koordinatları — örnek75c.html
- İlmeç koordinatları — örnek75d.html
- OnMouseOver olayı — örnek75e.html
- OnMouseOver olayı — örnek75f.html
- Hangi olay tetiklendi? — örnek75g.html
- Bubbling evet oluştu mu? — örnek75h.html
- Cancelable evet oluştu mu? — örnek75k.html
- Fare sağ tuşu devre dışı — örnek75p.html

67

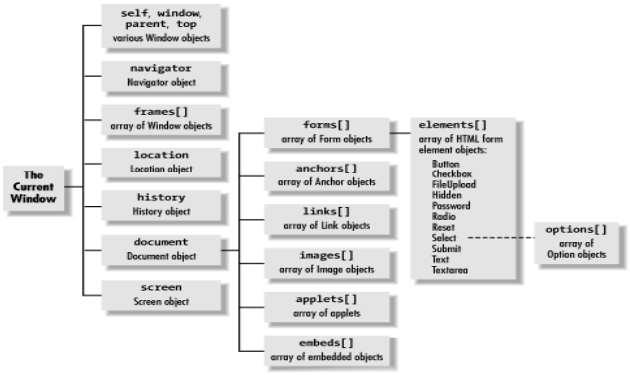
Document Object Model - DOM

- Javascript ile yazacağımız programlar, Netscape veya Internet Explorer programlarının belge nesne modeli (Document Object Model) denen kurallar içinde hareket etmek zorundadır.
- Daha yüksek programlama dillerine, örneğin C++, Delphi veya Java gibi dillerle program yazmışsanız, programcı olarak başka bir programın modeli ile sınırlı olmadığınızı, hatta işletim sisteminin programlar için sağladığı arayüzle (API) kendinizi sınırlı hissetmezsiniz.
- Fakat Javascript dahil tüm Script dilleri, Browser'ın sunduğu hiyerarşik nesne modeli ile sınırlıdır.

```
window.document.forms[0]
window.document.forms[0].elements[0]
```

68

Document Object Model - DOM



69

JavaScript HTML DOM nesneleri

- JavaScript nesnelerine ek olarak HTML DOM nesnelerine de erişebilirsiniz.
- `Window` : JavaScript içerisindeki en üst seviye nesnedir. Bu nesne tarayıcı penceresi ile ifade edilir. `<body>` veya `<frameset>` takılarının bir örneğiyle birlikte otomatik olarak oluşturulur.
- `Navigator` : kullanıcı tarayıcısı hakkındaki bilgileri içerir.
- `Screen` : kullanıcı ekranı hakkındaki bilgileri içerir.
- `History` : Kullanıcı tarayıcısının ziyaret ettiği URL bilgisini içerir.
- `Location` : O andaki URL hakkındaki bilgileri içerir.

70

getElementById() and getElementsByTagName()

- HTML dokümanı içerisindeki bir elemana erişmek için kullanılır.
- `getElementById()` metodu `ID` si belirtilmiş olan elemanı geri döndürür.

```
document.getElementById("someID");
```

html örnekler\örnek55.html

- Örneğin, `getElementsByTagName()` metodu doküman içerindeki yapıyı ihmal ederek içerisindeki tüm `<p>` elemanlarını bulabilir.

```
document.getElementsByTagName("tagname");
```

html örnekler\örnek55.html

71

window nesnesi

- `window` nesnesi en üst düzeyli nesne olduğu için özellikleri ve metodların başlarına `window.` yazmaya gerek yoktur.
- Frame ve Status Bar window nesnesinin özellikleridir. Status Bar kontrolü

```
window.status="Merhaba Dünya";
```

gibi bir kodla kontrol edilebilir.

- Ayrıca window nesnesinin
  - alert(), örnek50.html
  - prompt(), örnek49.html
  - confirm() örnek48.html
  - open()

gibi (pop up pencereleri vardır) metotları vardır.

72

window.open

- `window.open("html", "name",` özellikler) komutundaki özellikler ise:
  - `"html"` gösterilmesini istediğimiz sayfa
  - `"name"` istediğimiz bir başlık
- özellikler
  - `"toolbar"` toolbar'ın gösterilme özelliği (yes/no ya da 0/1 olarak belirtilir).
  - `"status"` statusbar'ın gösterilme özelliği (yes/no ya da 0/1 olarak belirtilir).
  - `"menubar"` menubar'ın gösterilme özelliği (yes/no ya da 0/1 olarak belirtilir).
  - `"scrollbars"` scrollbar denilen sayfayı aşağı-yukarı ve sağa-sola oynatmamızı sağlayan barların gösterilme özelliği (yes/no ya da 0/1 olarak belirtilir).
  - `"resizable"` açılacak olan ekranın boyutunun değiştirilebilir olup olmama özelliği (yes/no ya da 0/1 olarak belirtilir).
  - `"width"` genişlik (pixel olarak belirlenir).
  - `"height"` yükseklik (pixel olarak belirlenir).
- html örnekler\örnek47.html

73

window nesnesi

- Açık olan pencerenin sol üst köşe koordinatları değiştirilerek hareket ettirilmesi için `moveTo( )` kullanılır.
- `moveBy( )` ile pencere sağa, sola, yukarı ve aşağıya istenildiği kadar pixel hareket ettirilir.
- `resizeTo( )` ve `resizeBy( )` ile pencerenin yeniden boyutlandırmasını yapabilirsiniz
- window veya frame in sağa, sola, yukarı ve aşağıya istenildiği kadar pixel kaydırılmasını `scrollBy( )` metodu ile yapabilirsiniz.
- html örnekler\örnek52a.html

74

Navigator Nesnesi

- Tarayıcı hakkında bilgiler görüntülememizi sağlar:
- `appName` : Tarayıcının ismi
- `appVersion` : Tarayıc versiyonu
- `userAgent` : Taryıcı ismi ve versiyonu bilgilerinin her ikisini sağlar.
- `appCodeName` : Tarayıcının kod ismi, Netscape için Mozilla, Internet explorer için IE.
- `platform` : Tarayıcının çalıştığı platform.

```
var browser = "BROWSER INFORMATION:\n";
for(var propname in navigator) {
      browser += propname + ": " + navigator[propname] + "\n"
}
alert(browser);
```

html örnekler\örnek51.html

75

Screen Nesnesi

- Tarayıcının çalışmakta olduğu ekran hakkında bilgiler sunar.
- Pixel çinsinden ekranın genişliği ve yüksekliği, `screen.width` ve `screen.height` ile belirlenir.
- `availWidth` and `availHeight` özelikleriyle de kullanılabilir ekran genişlik ve yüksekli hakkında bilgi alınır.
- Ekranın desteklediği renk derinliği `screen.colorDepth` özelliği ile belirlenir.
- html örnekler\örnek52.html html örnekler\örnek53.html

76

History

Özellik	Tanım
length	history listesindeki eleman sayısını verir

Methot	Tanım
back()	history listesindeki bir önceki URL yi yükler
forward()	history listesindeki bir sonraki URL yi yükler
go(-2)	history listesinde istenilen URL ye gider.

77

Location

Özellik	Tanım
hash	Hash (#) işareti ayarlar veya geri gönderir.
host	O andaki URL hostname ve port numarısını verir.
hostname	URL nin hostname'ini verir.
href	URL nin tamamını geri gönderir.
pathname	URL yolunu ayarlar veya geri getirir.
port	URL nin port numarasını ayarlar veya geri getirir.
protocol	URL protokolünü geri getirir.
search	(?) soru işaretinden URL ayarlar veya geri getirir.

örnek54.html
örnek54a.html
örnek54b.html

Methot	Tanım
assign()	Yeni sayfa yükler
reload()	Sayfayı tekrar yükler
replace()	Sayfanın yerine yenisini yükler

78

style

- Bir nesneye ait özelliklerin ayarlanmasına veya bu özelliğin değerinin geri getirilmesine imkan sağlar.

```
document.getElementById("id").style.property="value"
```

- style nesnesi alt kategorileri:
  - Background
  - Border and Margin
  - Layout
  - List
  - Misc
  - Positioning
  - Printing
  - Scrollbar
  - Table
  - Text
  - Standard

Background özellikleri

Property	Description
background	Sets all background properties in one
backgroundAttachment	Sets whether a background-image is fixed or scrolls with the page
backgroundColor	Sets the background-color of an element
backgroundImage	Sets the background-image of an element
backgroundPosition	Sets the starting position of a background-image
backgroundPositionX	Sets the x-coordinates of the backgroundPosition property
backgroundPositionY	Sets the y-coordinates of the backgroundPosition property
backgroundRepeat	Sets if/how a background-image will be repeated

cookies

- Cookies ziyaretçinin bilgisayarında depolanan bir değişkendir.
- Bilgisayar tarayıcı işle bir sayfa istediğinde, cookie de gönderilecektir.
- JavaScrip ile hem cookie oluşturabilir hem de cookie değerlerini geri alabilirsiniz.
- Cookie örnekleri:
  - İsim cookie : ziyaretçi web sayfanızı ziyaret ettiğinde isminin girilmesi istenebilir. Bu isim bir cookie içerisinde depolanır. Daha sonra aynı ziyaretci tekrar sayfanızı açtığında ona "Merhaba ibrahim" şeklinde cookie içerisindeki ismi okuyarak güzel bir karşılama yapabilirsiniz.
  - Password cookie: Ziyaretçi sayfanızı ziyaret ettiğinde bir kullanıcı adı ve şifre girmesi istenir. Kullanıcı adı ve şifre bir cookie içerisine depolanır ve gelecek seferde kişi aynı sayfaya girdiğinde bu değerler otomatik kullanılarak bu işlemler tekrarlanmaz.
  - Date cookie: Ziyaretçi sayfanızı ziyaret ettiğinde o anki tarih cookie içerisinide depolanır. Daha sonraki ziyarette kişinin kaç gün önce zyarette bulunduğu bir mesaj ile cookie deki tarih değeri geri çağrılarak bir mesaj halinde sunulabilir.

Cookie oluşturmak

```
function setCookie(c_name,value,expiredays) {
  var exdate=new Date()
  exdate.setDate(exdate.getDate()+expiredays)
  document.cookie=c_name+ "=" +escape(value)+
      ((expiredays==null) ? "" :
      ";expires="+exdate.toGMTString())
}
```

`escape(string)` fonksiyonu string ifadesi içerisindeki karakterleri bütün bilgisayarların anlayabileceği kodlara dönüştürür. Bunun etkisi özel semboller üzerinde görülür.

```
escape("?!=()#%&"))
%3F%21%3D%28%29%23%25%26
```


Cookie değerleri çağırmak

```
function getCookie(c_name) {
  if (document.cookie.length>0) {
    c_start=document.cookie.indexOf(c_name + "=")
    if (c_start!=-1) {
      c_start=c_start + c_name.length+1
      c_end=document.cookie.indexOf(";",c_start)
      if (c_end==-1)
        c_end=document.cookie.length
        return
            unescape(document.cookie.substring(c_start,c_end));
      }
    }
  return ""
}
```


Cookie değeri kontrol etmek

```
function checkCookie() {
  username=getCookie('username')
  if (username!=null && username!="") {
      alert('Welcome again '+username+'!')}
  else {
      username=prompt('Please enter your name:',"")
      if (username!=null && username!="") {
                  setCookie('username',username,365)
      }
  }
}
```

html örnekler\örnek56.html

Düğme Grafiği Animasyonu

- HTML'in Anchor etiketi, Mouse işaretçisinin üzerine gelmesi (MouseOver) ve geri çekilmesi (MouseOut) olaylarına karşı tepki verebilmektedir.
- Biz iki olayı onMouseOver ve onMouseOut yönlendiricileri ile istediğimiz fonksiyona bağlayabiliyorduk.
- Önce fonksiyonumuzda kullanmak üzere dört değişken tanımlıyoruz ve bunlara düğmelerimizin adlarını ve boyutlarını değer olarak veriyoruz.

85

Düğme Grafiği Animasyonu

- Bu alıştırmayı yapmaya başlamadan önce aynı boyda dört grafiğiniz olması gerekir:
  - (1) İleri düğmesinin Mouse işaretçisi üzerine geldiği sıradaki görüntüsü (ileri_on.gif),
  - (2) İleri düğmesinin Mouse işaretçisi üzerinden çekildiği sıradaki görüntüsü (ileri_out.gif),
  - (3) Geri düğmesinin Mouse işaretçisi üzerine geldiği sıradaki görüntüsü (geri_on.gif),
  - (4) Geri düğmesinin Mouse işaretçisi üzerinden çekildiği sıradaki görüntüsü (geri_out.gif).
- Konumuz grafikçilik değil, fakat düğme grafiklerinizde yazı ve diğer unsurların yerlerinin aynı olması ve "on" ve "out" türleri arasında dikkat çekici bir derinlik (boyut) farkı bulunması yerinde olur.

html örnekler\örnek85.html

86

Katman Tekniği (DIV, LAYER)

- DIV veya LAYER etiketi ile oluşturacağınız katman ile HTML sayfasının üzerine, altını gösteren bir parşömen kat koymuş oluyorsunuz.
- CSS tekniklerini kullanarak Layer özelliklerini sayfada diğer herhangi bir HTML unsurundan daha büyük ölçüde ve çok daha hassas şekilde biçimlendirebilir; hatta hareket özelliği kazandırabilirsiniz.
- Javascript, katmanların bu özelliklerini, çeşitli olaylardan yararlanarak, olay-yönlendiricilerine ve metotlara bağlayabilirsiniz.
- Katman oluşturmakta kullanabileceğimiz etiketlerden biri olan LAYER Netscape tarafından tanınır fakat IE tarafından tanımaz.

87

Katman Tekniği (DIV, LAYER)

- Bu sebeple sayfalarımızda katmanları LAYER yerine DIV ile oluşturarak, iki Browser'u da kullanabiliriz.
- DIV etiketi de LAYER gibi katmanlar yapar; özellikleri ve metotları da LAYER gibidir. Aralarındaki tek fark, DIV ile oluşturulacak katmanların biçim ve konum özellikleri kendi STYLE komutları ile kazandırmak zorundasınız; oysa LAYER'in çok daha kestirme kendi biçim özellikleri vardır.
- Bununla birlikte STYLE metodunu kullanmakla DIV etiketine Javascript ile biçimlendirilecek daha çok "özellik" kazandırabiliriz.

88

DIV

<DIV

- **ALIGN=CENTER, LEFT, RIGHT** Sayfada veya tablo içinde bulunduğu yere göre, ortaya, sola veya sağa hizalanmasını sağlar.
- **CLASS=sinif_adi** Uygulanan stil sınıfı varsa, burada belirterek bütün DIV'in aynı stili almasını sağlarız.
- **DATAFLD=sütun_adi** DIV'e bir veritabanının değeri veriliyor, verilerin geleceği sütunun adi burada belirtilir.
- **DATAFORMATAS=HTML, TEXT** Bağlanan veritabanının HTML olarak mi, yoksa düz yazı olarak mi yorumlanacağını belirtmeye yarar.
- **DATASRC=#ID** Varsa, bağlanan veritabanının kimliği
- **ID=değer** Bu DIV'in kimliği

89

DIV

- **LANG=dil** ISO standartlarına göre bu bölümde yer alacak metnin yorumlanmasında uygulanacak dil kodu
- **LANGUAGE=dil** JAVASCRiPT, JSCRiPT, VBS veya VBSCRiPT. Bu DiV etiketinin içindeki Script'in dili. Hiç bir değer belirtmezseniz, JavaScript varsayılır.
- **STYLE=css1-özellikleri** Bu etiketin unsurlarına uygulanacak stil komutları
- **TITLE=başlık** Bilgi için kullanılır; Bu unsurun değeri onMouseOver halinde araç bilgi notu olarak görüntülenir.

> ... </DIV>

html örnekler\örnek60.html

90

Örnekler

- Dikey Kayan menüler örnek76.html
- Yatay Kayan menüler örnek76a.html
- Drop down menüler örnek76b.html
- Sayfa açılışı örnek76c.html
- Sayfa açılışı örnek76d.html
- Düğme arka plan resmi örnek76e.html
- Textarea arka plan rengi örnek76f.html

91

JavaScript ve Framler

- Frame kullanılan sayfalar en az üç web sayfasından oluşur.
- Frameleri içeren ilk sayfa frameset sayfası olarak adlandırılır.
- Framesetin içerisindeki diğer frameler child olarak adlandırılır.
- Uygulamalarda, her bir child için farklı isimler kullanılır.
- Framesetler ise top veya parent olarak isimlendirilir.
- Parent framedeki en üst seviye görüntü seviyesidir.
- Child pencereler parent pencereler içerisinde ortaya çıkar.

92

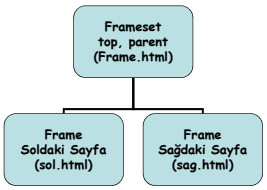
JavaScript ve Framler

```
<html>
<head>
<title>Çerçeveler</title>
</head>

<frameset cols="20%,80%">
   <frame name="sol" src="sol.html">
   <frame name="sag" src="sag.html">
</frameset>

<noframes> <body></body> </noframes>
</html>
```

frame.html



93

Bir Çerçeveden Diğerini Kontrol

```
<html>
<head>
<title>Çerçeveler</title>
</head>

<frameset rows="40%,60%">
    <frame name="ust" src="ust1.html">
    <frameset cols="40%,60%,*">
          <frame name="sol" src="sol1.html">
          <frame name="sag" src="sag1.html">
    </frameset>
</frameset>

<noframes><body></body></noframes>
</html>
```

Frame1.html

```
<script language="javascript">
fonrengi= new Array("red", "blue", "green",
      "yellow", "black", "navy", "teal", "silver")

function renkkendi(){
      restgelesayi= Math.floor(8*Math.random());
      document.bgColor = fonrengi[restgelesayi];
}

function renkust(){
      restgelesayi= Math.floor(8*Math.random());
      parent.ust.document.bgColor =
fonrengi[restgelesayi];
}


</script>
```

94