SAKARYA ÜNİVERSİTESİ

Web Teknolojileri
JavaScript
2/2

Dr. Öğretim Üyesi Gülüzar ÇİT
Dr. Öğretim Üyesi Hüseyin ESKİ
Öğr. Gör. Dr. Can YÜZKOLLAR

Konu & İçerik

- JavaScript

JavaScript Nesneleri

- Objeler(Nesneler) gerçek hayattaki varlıkları modelleyen değişkenlerdir.
- Örnek olarak bir arabayı obje olarak modellersek;
  - Bir arabanın ağırlık, renk gibi özellikleri varken çalıştır ve durdur şeklinde metotları bulunmaktadır.

JavaScript Nesneleri…

- obje ⇒ araba

Özellik	Metod
araba.marka=Fiat	araba.calistir()
araba.model=500	araba.sur()
araba.agirlik=850kg	araba.rolanti()
araba.renk=beyaz	araba.durdur()

- Tanımlanan bir nesne ile birden fazla araba oluşturabilir.
- Oluşturulan her araba farklı özelliklere sahip olabilir.
- Örneğin bir arabanın rengi siyahken diğer beyaz olabilir.

JavaScript Nesneleri...

- Aşağıdaki kodda **araba** değişkenine basit bir Fiat string değeri aktarıldı.

```
var araba = "Fiat";
```

- İkinci kodda araba değişkenine birden fazla değer aktarılmaktadır.

```
var araba = {marka:"Fiat", model:"500", renk:"beyaz"};
```

- JavaScript nesnelerine **isim:değer** şeklinde değer aktarılır. Virgüllerle birden fazla değer birbirinden ayrılır.

```
var kisi = {ad:"Ayşe", soyad:"Yılmaz", yas:50, gozRengi:"mavi"};
```

JavaScript Nesneleri…

- Nesne özelliklerine iki farklı yolla erişilebilir:

nesneAdı.özellikAdı veya ***nesneAdı["özellikAdı"]***

```
var kisi = {ad:"Ayşe", soyad:"Yılmaz", yas:50, gozRengi:"mavi"};
…
// Nesne özelliğine erişme
kisi.soyad;            //1. yol
kisi["soyad"];         //2. yol
```

JavaScript Nesneleri…

- Nesne metotlarına erişim ***nesneAdı.metotAdı();***

```
var kisi = {
  ad: "Ahmet",
  soyad : "Yılmaz",
  id        : 5566,
  adSoyad : function() {
    return this.ad + " " + this.soyad;
  }
};
…
var isim=kisi.adSoyad();      //DOĞRU  ERİŞİM
var isim=kisi.adSoyad;        //YANLIŞ ERİŞİM
```

JavaScript Nesneleri…

- **ÖRNEK:** Nesne özellik erişimi

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Sonucu Burada Göster…</p>
<script>
  var kisi = {ad: "Ahmet", soyad : "Yılmaz"};
   document.getElementById("demo").innerHTML = kisi.ad;
</script>
</body>
</html>
```

JavaScript Nesneleri…

- **ÖRNEK:** Nesne metot erişimi

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Sonucu Burada Göster…</p>
<script>
 var kisi = {
    ad: "Ahmet",
    soyad : "Yılmaz",
    id       : 5566,
    adSoyad : function() {
       return this.ad + " " + this.soyad;} };
  document.getElementById("demo").innerHTML = kisi.adSoyad();
</script>
</body>
</html>
```

Değişkenlerin Yaşam Alanları

- Değişkenler yaşam alanlarına göre iki türdedir.
  - Global değişkenler
  - Lokal değişkenler

- Global değişkenlere javascript kodunda her yerden ulaşılabilir ve yaşam alanları tüm javascript kodu kadardır.

- Lokal değişkenler ise sadece tanımlandığı bloktan erişilebilir ve yaşam alanları blok içindedir. Bloğun dışına çıkılınca ölürler.

Değişkenlerin Yaşam Alanları…

- **Global Değişkenler**
  - Global değişkenlere javascript kodunda her yerden ulaşılabilir ve yaşam alanları tüm javascript kodu kadardır.

```
var araba = " Volvo";
…
// araba değişkenine buradan erişilebilir.

function myFunction() {
    …
 // araba değişkenine buradan da erişilebilir
}
```

Değişkenlerin Yaşam Alanları…

- **Global Değişkenler…**
  - **ÖRNEK:**

```
<!DOCTYPE html>
<html>
<body>
<p>Bir GLOBAL değişken herhangi bir script veya fonksiyonda
   kullanılabilir</p>
<p id="demo">Sonucu Burada Göster…</p>
<script>
    var araba = "Volvo";
  myFunction();

  function myFunction() {
     document.getElementById("demo").innerHTML = "Bu araba" + araba;
  }
</script>
</body>
</html>
```

Değişkenlerin Yaşam Alanları…

- **Lokal Değişkenler**
  - Lokal değişkenler ise sadece tanımlandığı bloktan erişilebilir ve yaşam alanları blok içindedir. Bloğun dışına çıkılınca ölürler.

```
// araba değişkenine buradan ulaşılamaz.
// Burada yaşamamaktadır.
function myFunction() {
    var araba = "Volvo";
    …
    // araba değişkenine buradan erişilebilir.
    // Değişken bu blok için yaşar.
}
```

Değişkenlerin Yaşam Alanları…

- **Lokal Değişkenler…**
  - **ÖRNEK:**

```
<!DOCTYPE html>
<html>
<body>
<p>Bir LOKAL değişken sadece tanımlandığı fonksiyon içerisinden
   erişilebilmektedir</p>
<p id="demo">Sonucu Burada Göster…</p>
<script>
   myFunction();
   document.getElementById("demo").innerHTML="Bu araba" + typeof araba;

   function myFunction() {
     var araba = "Volvo";
   }
</script>
</body>
</html>
```

JavaScript String

- Stringler metinsel bilgileri saklayan değişken tipleridir.
- İki şekilde tanımlanabilir; tek tırnak veya çift tırnak içinde.

```
var araba = "Volvo XC60";
var araba = 'Volvo XC60';
```

- String uzunluğu length özelliği ile bulunur.

```
var metin = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
var sln = metin.length;
```

- String özel karekterleri escape karetkeri ile yazdırabilirsiniz.

```
var x = 'It\'s alright';
var y = "\"Sakarya Üniversitesi\"ne Hoşgeldiniz."
```

JavaScript String…

Özel Karakterler

KOD	ÇIKTI
\'	Tek tırnak
\"	Çift tırnak
\\	backslash
\n	Yeni satır
\r	Satır başı
\t	tab
\b	backspace
\f	Sayfa başı

JavaScript String…

String Metotları

METOT	AÇIKLAMA
charAt()	İndeksi verilen karakteri döndürür(pozisyon)
indexOf()	String'te verilen değerin ilk bulunduğu indisi döndürür
concat()	İki veya daha fazla string'i birleştirir ve birleşmiş string'i döndürür
lastIndexOf()	Stringte verilen değerin son bulunduğu indisi döndürür
replace()	Stringte yer bir metni bulup değiştirmek için kullanılır
search()	String içinde bir metnin pozisyonunu döndürür
slice()	Metinde Başlangıç ve bitiş değeri verilen aralıktaki metni döndürür
split()	verilen karaktere göre metni bölerek diziye dönüştürür
substr()	Metinde Başlangıç ve uzunluğu verilen aralıktaki metni döndürür
substring()	Metinde Başlangıç ve bitiş değeri verilen aralıktaki metni döndürür

JavaScript String…

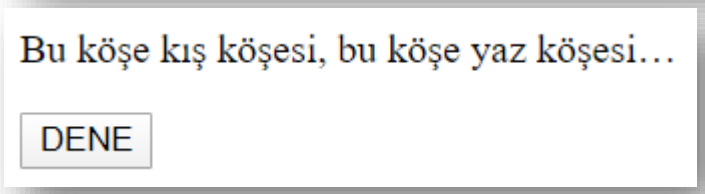
String Metotları…

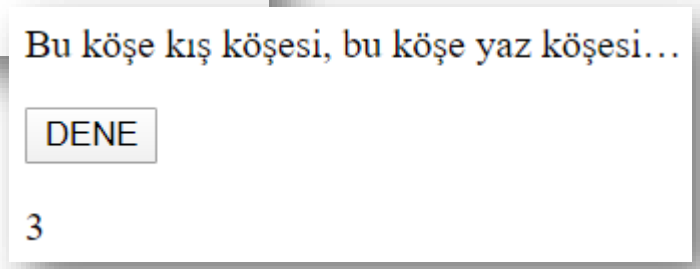
METOT	AÇIKLAMA
toLocaleLowerCase()	Sunucunun bölge ayarlarına referans alarak metni küçük harfe çevirir
toLocaleUpperCase()	Sunucunun bölge ayarlarına referans alarak metni büyük harfe çevirir
toLowerCase()	Metni küçük harfe çevirir
toString()	Nesneyi string ifadeye dönüştürür
toUpperCase()	Metni büyük harfe çevirir
trim()	Metnin başındaki ve sonundaki boşlukları siler
valueOf()	Nesnenin string değerini döndürür

JavaScript String…

ÖRNEK:

```
<!DOCTYPE html>
<html>
<body>
<p id="p1">Bu köşe kış köşesi, bu köşe yaz köşesi…</p>
<button onclick="myFunction()">DENE</button>
<p id="demo"></p>
<script>
   function myFunction() {
      var str = document.getElementById("p1").innerHTML;
      var pos = str.indexOf("köşe");
    document.getElementById("demo").innerHTML = pos;
}
</script>
</body>
</html>
```





JavaScript String…

ÖRNEK:

```
<!DOCTYPE html>
<html>
<body>

<p>substr() metodu, bir karakter katarının bir parçasını çıkartır ve
çıkartılan parçayı yeni bir karakter katarında döndürür:</p>
<p id="demo"></p>

<script>
var str = "Elma, Muz, Kivi";
document.getElementById("demo").innerHTML = str.substring(6,10);
</script>

</body>
</html>
```

substr() metodu, bir karakter katarının bir parçasını çıkartır ve çıkartılan parçayı yeni bir karakter katarında döndürür:

Muz,

JavaScript Number

- **parseInt** tamsayı tipine dönüştürür.

```
parseInt("10");          // 10 döndürür
parseInt("10.33");       // 10 döndürür
parseInt("10 20 30");    // 10 döndürür
parseInt("10 yıl");      // 10 döndürür
parseInt("yıl 10");      // NaN döndürür
```

- **parseFloat()** ondalık sayı tipine dönüştürür.

```
parseFloat("10");          // 10 döndürür
parseFloat("10.33");       // 10.33 döndürür
parseFloat("10 20 30");    // 10 döndürür
parseFloat("10 yıl");      // 10 döndürür
parseFloat("yıl 10");      // NaN döndürür
```

JavaScript Number

- **valueOf()** sayısal değerini geri gönderir.

```
var x = 123;
x.valueOf();              // x değişkeninden 123 döndürür
(123).valueOf();          // tamsayı 123 den 123 döndürür
(100 + 23).valueOf();     // 100+23 ifadesinden 123 döndürür
```

JavaScript Tarih&Saat

```
new Date()
new Date(milliseconds)   //86400000
new Date(dateString)     //"October 13, 2014 11:13:00"
new Date(year, month, day, hours, minutes, seconds, milliseconds)
                         //99, 5, 24, 11, 33, 30, 0
```

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var d = new Date();
document.getElementById("demo").innerHTML = d;
</script>

</body>
</html>
```

Wed Apr 29 2020 16:01:58 GMT+0300 (GMT+03:00)

JavaScript Diziler

Söz Dizimi (Syntax)

```
var dizi-adi = [item1, item2, ...];
```

ÖRNEK:

```
<!DOCTYPE html>
<html>
<body>

<p id="demo1"></p>
<p id="demo2"></p>

<script>
   var arabalar = ["Saab","Volvo","BMW"];
   document.getElementById("demo1").innerHTML = arabalar;
   document.getElementById("demo2").innerHTML = arabalar[0];
</script>

</body>
</html>
```

Saab,Volvo,BMW

Saab

JavaScript Diziler…

- Farklı veri tipleri tek bir dizide barındırılabilir.

```
var kisi = ["Ayşe", "Yılmaz", 46];
```

- **length()** özelliği ile dizinin uzunluğu bulunabilir.

```
var meyveler = ["Muz", "Portakal", "Elma", "Mango"];
meyveler.length;    // meyveler dizisinin uzunluğu:4
```

- Dizi elemanlarında döngü yardımıyla dolaşılabilir.

```
var metin="";
var index;
var meyveler = ["Muz", "Portakal", "Elma", "Mango"];
for (index = 0; index < meyveler.length; index++) {
    metin += meyveler[index];
}
```

JavaScript Diziler…

- Dizilere iki yolla eleman eklenebilir.
  - Son eleman olarak ekleme

```
var meyveler = ["Muz", "Portakal", "Elma", "Mango"];
meyveler[meyveler.length] = "Limon"; //meyveler dizisine
                                     //(Limon) ekleme
```

  - **push** metoduyla ekleme

```
var meyveler = ["Muz", "Portakal", "Elma", "Mango"];
meyveler.push("Limon");   //meyveler dizisine (Limon) ekleme
```

Javascript Boolean

- İki değere sahip olan programlamada sıklıkla kullanılan değişken tipidir.

YES / NO
ON / OFF
TRUE / FALSE

- **Boolean()** fonksiyonu karşılaştırmanın sonucunu verir.

```
Boolean(10 > 9)          // true döndürür
```

- 0 değeri **false** olarak değerlendirilir.

```
var x = 0;
Boolean(x);              // false döndürür
```

Karşılaştırma Operatörleri

OPERATÖR	AÇIKLAMA	KARŞILAŞTIRMA	DÖNEN DEĞER
==	eşit	x == 8	false
		x == 5	true
		x == "5"	true
===	değer ve tipi eşit	x === 5	true
		x === "5"	false
!=	eşit değil	x != 8	true
!==	değeri veya tipi eşit değil	x !== 5	false
		x !== "5"	true
		x !== 8	true
>	büyük	x > 8	false
<	küçük	x < 8	true
>=	Büyük veya eşit	x >= 8	false
<=	Küçük veya eşit	x <= 8	true

```
var x = 5;
```

Mantıksal Operatörler

OPERATÖR	AÇIKLAMA	ÖRNEK
&&	AND	(x < 10 && y > 1) is true
||	OR	(x == 5 || y == 5) is false
!	NOT	!(x == y) is true

```
var x = 5;
var y = 5;
```

Karşılaştırma

Söz Dizimi (Syntax)

```
if (koşul) {
    koşul doğruysa yapılacaklar
}
```

ÖRNEK:

```
if (saat < 18) {
    mesaj = "İyi Günler…";
}
```

Karşılaştırma…

- **Söz Dizimi (Syntax)**

```
if (koşul) {
    koşul doğruysa yapılacaklar
} else {
    koşul yanlışsa yapılacaklar
}
```

- **ÖRNEK:**

```
if (saat < 18) {
    mesaj = "İyi Günler…";
} else {
    mesaj = "İyi Akşamlar…";
}
```

Karşılaştırma…

Söz Dizimi (Syntax)

```
if (koşul1) {
    koşul1 doğruysa yapılacaklar
} else if (koşul2) {
    koşul1 yanlış ve koşul2 doğruysa yapılacaklar
} else {
    koşul1 ve koşul2 yanlışsa yapılacaklar
}
```

ÖRNEK:

```
if (saat < 10)          {mesaj = "Günaydın…";}
else if (saat < 20)     {mesaj = "İyi Günler…";}
else                    {mesaj = "İyi Akşamlar…";}
```

Karşılaştırma…

➢ **ÖRNEK:**

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Sonucu Burada Göster.</p>
<script>
   var mesaj;
   var saat = new Date().getHours();
   if (saat < 18) {
     mesaj = "İyi Günler…";
   }
   else {
      mesaj = "İyi Akşamlar…";
   }
   document.getElementById("demo").innerHTML = mesaj;
</script>

</body>
</html>
```

İyi Akşamlar…

Switch

Söz Dizimi (Syntax)

```
switch(ifade) {
    case n:
        kod bloğu
        break;
    case n:
        kod bloğu
        break;
    default:
        varsayılan kod bloğu
}
```

Switch…

- **ÖRNEK:**

```
switch (new Date().getDay()) {
    case 0: gun = "Pazar";          break;
    case 1: gun = "Pazartesi";      break;
    case 2: gun = "Salı";           break;
    case 3: gun = "Çarşamba";       break;
    case 4: gun = "Perşembe";       break;
    case 5: gun = "Cuma";           break;
    case 6: gun = "Cumartesi";      break;
    default: alert("!!!");          break;
}
```

Döngüler – FOR

Söz Dizimi (Syntax)

```
for (ifade 1; ifade 2; ifade 3) {
    gerçekleştirilecek kod bloğu
}
```

ÖRNEK:

```
for (i = 0; i < 5; i++) {
    text += "Sayı " + i + "<br>";
}
```

```
for (i = 0, len = arabalar.length, metin = ""; i < len; i++)
{
    metin += arabalar[i] + "<br>";
}
```

Döngüler – FOR…

➢ **ÖRNEK:**

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>
<script>
   var metin = "";
   var i;
   for (i = 1; i < 10; i = i + 2) {
       metin += i + "<br>";
   }
   document.getElementById("demo").innerHTML = metin;
</script>

</body>
</html>
```

1
3
5
7
9

Döngüler – WHILE

Söz Dizimi (Syntax)

```
while (koşul) {
    gerçekleştirilecek kod bloğu
}
```

ÖRNEK:

```
while (i < 10) {
    text += "The number is " + i;
    i++;
}
```

Döngüler – DO…WHILE

Söz Dizimi (Syntax)

```
do {
    gerçekleştirilecek kod bloğu
}
while (koşul);
```

ÖRNEK:

```
do {
    text += "Sayı " + i;
    i++;
}
while (i < 10);
```

Break & Continue

- **break** komutu döngüyü kullanıldığı yerde kırarak sonlandırır.

```
for (i = 0; i < 10; i++) {
    if (i == 3) { break; }
    text += "Sayı:  " + i + "<br>";
}
```

- **continue** komutu döngüyü kullanıldığı yerde işlem yaptırmadan bir sonraki iterasyona yönlendirir.

```
for (i = 0; i < 10; i++) {
    if (i == 3) { continue; }
    text += "Sayı: " + i + "<br>";
}
```

JavaScript JSON

- JSON saklamak ve taşımak için kullanılan bir veri biçimidir.
- Genellikle veriler bir sunucudan web sayfasına gönderildiğinde kullanılır.
- JSON, programlama dilinden bağımsız olan XML'e alternatif olarak kullanılan javascript tabanlı veri değişim formatıdır.
- JSON'un amacı veri alış verişi yaparken daha küçük boyutlarda veri alıp göndermektir.
- Bu özellikleri sayesinde JSON ile çok hızlı web uygulamaları oluşturabilir.

JavaScript JSON…

- **JSON sözdizimi kuralları**
  - Veri ad / değer çiftleri içinde yazılır
  - Veri virgül ile ayrılır
  - Köşeli parantezler diziler tutar

```
{
"calisan":[
  {"ad":"Ayşe", "soyad":"Yılmaz"},
  {"ad":"Mehmet", "soyad":"Öztürk"},
  {"ad":"Ömer", "soyad":"Çetin"}]
}
```

JavaScript JSON…

- **JSON verisi**
  - Javascript nesne özellikleri gibi ad / değer çiftleri içinde yazılır.

```
"ad":"Ayşe"
```

- JSON nesneleri
  - JSON nesneleri küme parantezi içine yazılır.
  - Sadece JavaScript gibi nesneler birden fazla ad / değer çiftlerini içerebilir:

```
{"ad":"Ayşe", "soyad":"Yılmaz"}
```

JavaScript JSON…

- **JSON dizisi**
  - JSON nesne dizileri köşeli parantez içinde yazılır.
  - JavaScript gibi, bir JSON dizisi nesneler içerebilir:

```
{
"calisan":[
  {"ad":"Ayşe", "soyad":"Yılmaz"},
  {"ad":"Mehmet", "soyad":"Öztürk"},
  {"ad":"Ömer", "soyad":"Çetin"}]
}
```

JavaScript JSON…

- **Bir JSON metnini JavaScript nesnesine dönüştürme**
  - JSON'un genel kullanım amaçlarından birisi bir web sunucusundan veri okumak ve bu veriyi web sayfasında görüntülemektir.
  - Kolaylık içim, giriş olarak string kullanarak örneklenecektir.
  - İlk önce, JSON sözdizimi içeren bir Javascript string oluşturulur.

```
var metin = '{"calisan":[' +
   '{"ad":"Ayşe",  "soyad":"Yılmaz" },' +
   '{"ad":"Mehmet","soyad":"Öztürk" },' +
   '{"ad":"Ömer",  "soyad":"Çetin" }]
    }';
```

  - Daha sonra, string'i bir JavaScript nesnesine dönüştürmek için dahili fonksiyonu olan **JSON.parse()** kullanılır:

```
var nesne = JSON.parse(metin);
```

JavaScript JSON…

- **ÖRNEK:**

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>
<script>
   var metin = '{"calisan":[' +
   '{"ad":"Ayşe","soyad":"Yılmaz" },' +
   '{"ad":"Mehmet","soyad":"Öztürk" },' +
   '{"ad":"Ömer","soyad":"Çetin" }]}';
   obj = JSON.parse(metin);
   document.getElementById("demo").innerHTML =
   obj.calisan[1].ad + " " + obj.calisan[1].soyad;
</script>
</body>
</html>
```

Mehmet Öztürk

JavaScript JSON…

➢ ÖRNEK:

```
<!DOCTYPE html>
<html>
<body>

<table id="calisanlar">
  <tr><td>ADI</td><td>SOYADI</td><td>BÖLÜMÜ</td></tr>
</table>

<p id="demo"></p>
<script>
   var metin = '{"calisan":[' +
    '{"ad":"Ayşe","soyad":"Çetin", "bolumu":"İmalat" },' +
    '{"ad":"Mehmet","soyad":"Öztürk" , "bolumu":"Bilgisayar"},' +
    '{"ad":"Ömer","soyad":"Yılmaz" , "bolumu":"İmalat"}]}';
    personel = JSON.parse(metin);
    var tablo = document.getElementById("calisanlar");
    for (i  in personel.calisan) {
        var row = tablo.insertRow(-1);
        var cell1 = row.insertCell(0);
        var cell2 = row.insertCell(1);
        var cell3 = row.insertCell(2);
        cell1.innerHTML = personel.calisan[i].ad;
        cell2.innerHTML = personel.calisan[i].soyad;
        cell3.innerHTML = personel.calisan[i].bolumu;}
</script>
</body>
</html>
```

ADI	SOYADI	BÖLÜMÜ
Ayşe	Çetin	İmalat
Mehmet	Öztürk	Bilgisayar
Ömer	Yılmaz	İmalat

Kaynaklar

- http://www.w3schools.com/