JavaScript

Full Stack Yazılım Uzmanlığı Eğitimi

Haftalık Program

Değişkenler ve Sabitler

Operatörler

Tarih Fonksiyonları

Koşullu Yapılar

Döngüler

Fonksiyonlar

Diziler

for-of Döngüsü

for-in Döngüsü

Global Scope, Local Scope

Ternary Operator

Events, Event Listener

DOM Yapısı

Event Listener

DOM Element Add, Remove, Replace, InsertBefore

Değişkenler, Sabitler, Operatörler

```
// Değişken Tanımlama
var adi = "Siliconmade";
let asgariMaas = 5789.4789;
const personelSayisi = 10;

// Operatörler
let toplamOdeme = asgariMaas *
personelSayisi;

// Tarih
let odemeTarihi = new Date();

// null ve undefined türleri
let sayi1 = null;
//let sayi2 = undefined;
```

```
// Console panelini temizleme
console.clear();

// Console paneline bir bilgi yazdırma
console.log(adi);
console.info("Maaş = " + asgariMaas);
```

Değişkenler, Sabitler, Operatörler

```
// typeof: Bir değişkenin tipini
almayı sağlar.
// Genel tipler: string, number,
object

console.log(typeof adi);
console.log(typeof asgariMaas);
console.log(typeof personelSayisi);
console.log(typeof odemeTarihi);
console.log(typeof sayi1);
```

```
// Object Notation
// C# class yapısına benzer şekilde
birden fazla bilgiyi bir değişkende
object notation yapısıyla
saklayabiliriz.

var person = {
  name: 'Merve',
  birthYear: 2000
};
console.log(person.name);
```

Tarih Fonksiyonları

```
// Aktif tarih bilgisini alma
var bugun = new Date();

// Gün
console.log(bugun.getDate();)

// Ay
console.log(bugun.getMonth() + 1);

// Yıl
console.log(bugun.getFullYear());

// Saat
console.log(bugun.getHours());

// Dakika
console.log(bugun.getMinutes());

// Saniye
console.log((bugun.getSeconds());
```

Koşullu Yapılar

```
// if Yapısı: Bir şarta bağlı işlem yapmayı sağlar.

let uyeId = 3;
if (uyeId == 1) {
  console.log('1 numaralı üye');
} else if (uyeId == 2) {
  console.log('2 numaralı üye');
} else {
  console.log('Diğer üye');
}
```

```
// switch Yapısı: Bir değerin farklı durumlarına göre işlem yapmayı sağlar.
var haftaninGunu = new Date().getDay();
switch (haftaninGunu) {
  case 0:
    day = "Sunday";
    break;
  case 1:
    day = "Monday";
    break;
  case 2:
     day = "Tuesday";
    break;
  case 3:
    day = "Wednesday";
    break;
  case 4:
    day = "Thursday";
    break;
  case 5:
    day = "Friday";
    break;
  case 6:
    day = "Saturday";
    break;
  default:
    day = "Other"
    break;
}
console.log(day);
```

Soru: Bir gün değerinin sadece haftiçi veya haftasonu olmasını nasıl belirleriz?

Döngüler

// for Döngüsü: Belirli bir başlangıç değeri, bitiş koşulu ve artış miktarına göre bir işlemi döngüye alabiliriz.

```
for(let i = 1; i < 10; i++) {
  if (i == 2) continue;
  if (i == 6) break;
  console.log("Sayı = " + i);
}
```

let i = 1 → Başlangıç değeri
i < 10 → Bitiş koşulu
i++ → Artış miktarı
break → Döngüden çıkma
continue → Bir sonraki döngü değerine geçme

// while Döngüsü: Bir şarta göre çalışan döngü

```
let i = 0;
let str = '';
while (i < 9) {
  i++;
  if (i == 5) continue;
  str += i;
}
console.log(str);
```

i<9 → Bitiş koşulu

Uygulama

Belirtilen bir sayıya kadar olan çift sayıları fonksiyon kullanarak ekrana yazdıran program

Örneğin; bir değişken içerisinde sayi=20 olarak tanımlanırsa;

20'ye kadar olan tüm çift sayıları konsol ekranına yazdıracağız.

0 2 4 6 … 18 20 gibi alt alta

Uygulama

Çarpım Tablosu

Konsol ekranına sabit olarak tanımlanan bir sayıya ait çarpım tablosunu konsola yazdırın.

```
const sayi1 = 9;
```

Fonksiyonlar

```
// Fonksiyon Tanımı 1
function Carp(a, b) {
  return a * b;
}
var sonuc = Carp(2,5);

// Fonksiyon Tanımı 2
var Topla = function(a, b) {
  return a + b;
}
var sonuc = Topla(2,5);

// Arrow Function
var Topla = (a, b) => a + b;
```

```
// Fonksiyon Tanımı 3
var person = {
  name: 'Cem',
  birthYear: 1983,
  getAge: function(year) {
    //return 2022 - year;
    return (new Date().getFullYear())
- year;
  }
}

let year = person.birthYear;
console.log(person.getAge(year));
```

Diziler

```
let isimler = ["Cem", "Taylan",
"Dilay", "Emircan"];
console.log(isimler, isimler[3]);

let arr2 = ['AA', 12, { name: 'Cem' }]
console.log(arr2[2]);
console.log(arr2[2].name);

const fruits = ["apple", "orange",
"cherry"];
```

```
// Dizinin sonuna eleman ekleme
fruits.push('banana');
fruits.push("Kiwi", "Lemon");

// Son elemanı sil
fruits.pop();

// İlk elemanı sil
fruits.shift();

// Dizinin başına eleman ekleme
fruits.unshift("Lemon","Pineapple");
```

Diziler

```
// Diziden eleman silme: splice(index, silinecek eleman)
// 1. elemandan itibaren 1 eleman silme
fruits.splice(1, 1);

// Dizide araya elemen ekleme
fruits.splice(2, 0, "LemonA", "KiwiA");

// Dizideki eleman sayısı
console.log(fruits.length + " meyve var!");

// forEach Döngüsü
fruits.forEach(function(fruit, index) {
  console.log((index+1) + ". sıradaki eleman = " + fruit);
});
//console.log(fruits.indexOf('cherry2'));
```

```
// indexOf: dizideki içerdiği index
if (fruits.indexOf('cherry2') == -1) {
  console.log('Dizide cherry2 bulunmuyor!');
}

// includes: dizide içeriyorsa
if (!fruits.includes('cherry2')) {
  console.log('Dizide cherry2 bulunmuyor!');
}

// dizide varsa
if (fruits.includes('cherry')) {
  console.log('Dizide cherry var!');
}

// find: tek kayıt filtreleme
var orange = fruits.find(f => f == 'orange');
console.log(orange);

// filter: birden fazla kaydı filtreleme
var kiwis = fruits.filter(f => f == 'Kiwi');
console.log(kiwis);
```

Diziler: Sıralama

```
var isimler = ["Cem", "Emircan",
"İrem", "Ali", "Dilay", "Mert"];
console.log(isimler);
isimler.sort();
console.log(isimler);

var sayilar = [1, 12, 6, 5];
sayilar.sort(); // metinsel sıralama
yapar.
sayilar.sort(function(a, b) { return a
- b; }); // sayısal sıralama yapması
için
console.log(sayilar);
sayilar.reverse();
console.log(sayilar);
```

```
let yarisma = [
  { Isim: "Cem", Puan: 60 },
  { Isim: "Emircan", Puan: 65 },
  { Isim: "Dilay", Puan: 54 },
  { Isim: "Didem", Puan: 92 },
];
yarisma.sort(function(k1, k2) { return
k2.Puan - k1.Puan; });
console.log(yarisma);
console.log("En yuksek puan alan " +
yarisma[0].Isim + " in puanı " +
yarisma[0].Puan);
console.log("En dusuk puan alan " +
yarisma[3].Isim + " in puanı " +
yarisma[3].Puan);
console.log("En dusuk puan alan " +
yarisma[yarisma.length-1].Isim + " in puanı "
+ yarisma[yarisma.length-1].Puan);
```

Uygulama

Diziye eleman ekleme, en yüksek puan alan yarışmacıyı bulma

Aşağıdaki dizideki;

- Sanem isimli yarışmacıyı 87 puanla listeye ekleyelim.
- En yüksek puan alan yarışmacının adını yazdır
- En düşük puan alan yarışmacının adını yazdır
- 60 puan ve altındaki yarışmacıları eleyerek diziden kaldıralım.

```
let yarisma = [

  { Isim: "Cem", Puan: 60 },

  { Isim: "Emircan", Puan: 65 },

  { Isim: "Dilay", Puan: 54 },

  { Isim: "Didem", Puan: 92 },

];
```

Uygulama

Dizide JavaScript bilen kişileri yazdırma

Aşağıdaki dizideki;

- JavaScript bilen kişileri ekrana yazdırın.

```
let ogrenciler = [

  { isim: 'Cem', github_adresi: 'cemg', bildigi_diller: ['C#', 'ASP.NET'] },

  { isim: 'Ali', github_adresi: 'ali', bildigi_diller: ['C#'] },

  { isim: 'Emircan', github_adresi: 'emircan', bildigi_diller: ['C#', 'JavaScript'] },

  { isim: 'İrem', github_adresi: 'irem', bildigi_diller: ['C#', 'JavaScript']  },

];
```

for-of Döngüsü

foreach yapısındaki gibi dizi elemanlarını sırayla dolaşmayı sağlar.

```
const array1 = ['a', 'b', 'c'];

for (const element of array1) {
  console.log(element);
}
```

for-in Döngüsü

Obje elemanları arasında sırayla dolaşmayı sağlar.

```
const object = { a: 1, b: 2, c: 3 };

for (const property in object) {
  console.log(`${property}: ${object[property]}`);
}
```

Global Scope, Local Scope

```
{
  let x = 2;
}
console.log(x);
```

// let ile tanımlanırsa scope içerisinde yani süslü parantez içerisinde erişebildiği için ekranda bir çıktı vermez.

```
{
  var x = 2;
}
console.log(x);
```

// var ile tanımlanırsa scope bağımsız erişebildiği için ekranda bir çıktı verir.

Ternary Operator

```
var sayi = 2;
if (sayi>3) {
  console.log("A");
} else {
  console.log("B");
}

// yerine aşağıdaki gibi tek satırda kullanabiliriz.
console.log( sayi > 3 ? "A" : "B" );
console.log( sayi > 3 ? "A" : (sayi>1 ? "B" : "C") );
```

Events, Event Listener

Bir HTML elemanı üzerinde tıklanma, karaktere basma, fare durumlarında göre işlem yapmayı sağlar.
Bir event ile beraber bir işlem gerçekleşir.

Örnek event'lar: onclick, ondblclick, onmouseover, onkeypress, onkeydown, onkeyup

```
<button class="btn btn-primary"
onclick="alert('Merhaba')">Merhaba</button>
<button class="btn btn-primary"
onclick="uyariGoster()">Sil</button>
<button class="btn btn-primary"
onclick="sifreliGirisYap()">Şifreli
Giriş</button>
```

alert: Uyarı penceresi
confirm: Onay penceresi gösterir, true veya false döner
prompt: Bilgi girişi penceresi gösterir.

```
<script>
function uyariGoster() {
var secim = confirm('Silmek
istediğinize emin misin?');
        if (secim) {
                alert('Siliyorum
bak!');
        }
}
function sifreliGirisYap(e) {
        var girilenDeger =
prompt('Şifreniz');
        if (girilenDeger == '123') {
                alert('Doğru şifre
bildin!');
        }
}
</script>
```

DOM Yapısı

Document Object Model: Bir HTML belgesi içerisindeki her bir elemanın özelliklerini içeren yapıya DOM denilmektedir.

DOM içerisinde tüm elemanların özelliklerine erişim sağlayabilir veya bazı özelliklerini değiştirebiliriz.

Burada örneğin; sayfa içerisindeki window ile tarayıcı penceresine, document ile sayfa özelliklerine erişim sağlayabiliriz.

Örn:

```
window.document.title = 'Siliconmade';
document.body.style.backgroundColor =
'black';
```

document.getElementById

Standart HTML elemanları dışında sayfa içerisine eklediğimiz bir nesneye erişim sağlayarak DOM özelliklerini, metotlarını kullanabiliriz.

Burada görüldüğü gibi **getElementById** fonksiyonu ile DOM içerisindeki bir elemanı Id özelliğine göre seçmeyi sağlar.

Yani DOM içerisinde bir nesnenin bir özelliğine veya motoduna erişmek veya değiştirmek için önce seçiyoruz.

```
<input class="form-control" id="tahmin">
<div id="bilgi"></div>
<script>
let tahmin = document.getElementById("tahmin");
let girilenSayi = tahmin.value;
var bilgiElemani =
document.getElementById('bilgi');
bilgiElemani.innerHTML = "Merhaba Siliconmade
Academy';
bilgiElemani.style.color = "red";
bilgiElemani.style.fontSize = "20px";
</script>
```

DOM Yapısı

Seçtikten sonra bunu bir değişkene aktararak; **innerHTML** ile içerisindeki HTML değerini ayarlayabiliriz.

style özelliği stil özelliklerine erişim yaparak color ile yazı rengini ve **fontSize** ile yazı boyutunu ayarlamış olduk.

Görüldüğü gibi HTML DOM yapısı içerisinde her elemanın farklı özellikleri olabilir. Bu özelliklerin tamamını şu aşamada ezberlememize gerek yok. Zamanla uygulama yaptıkça öğrenebilir veya editörlerde nokta karakterinden sonra kod tamamlama yaparak da bazı özellikleri hatırlayabilirsiniz.

document.querySelector

```
let tahmin = document.querySelector("#tahmin"); //
ID elemanları # ile belirtiyoruz.
let cinsiyet =
document.querySelector("[name='cinsiyet']"); //
name değerine göre seçme
```

document.querySelectorAll → Birden fazla elemanı seçmeyi sağlar.

document.getElementsByName("cinsiyet") → name değerine göre seçmeyi sağlar.

Uygulama

Adı alanına girilen karakter kontrolü ve uzunluk gösterimi

Adı alanına girilen karakter kontrolü yaparak uzunluğunu metin kutusu altında bilgi alanında gösterin. 140 karakterden sonrasına giriş yapmasın.

Adınız:

```
<input type="text" name="adi"
id="adi"
onkeyup="karakterKontrol(event)">

<div id="bilgi"></div>
```

Uygulama

Sayı Bulma Oyunu

Bir sayı girişi yapacağımız bir metin kutusu olsun.

Global Scope'da 1-50 arasında rastgele sayı üretsin.

Metin kutusunun yanında yer alan bir Tahmin Et butonuna basıldığında ekranda yer alan bilgi ismindeki bir div kutusunda; Aşağı İn, Yukarı Çık ve Bildin şeklinde bilgilendirme yapsın.

Rastgele sayı üretme fonksiyonu:
Math.floor(Math.random() * 49) + 1

Math.random(): 0-1 arasında rastgele bir sayı üretir.

Event Listener

```
<button id="btn1" class="btn btn-warning">
    Tıkla
</button>


<script>
var btn1 = document.getElementById("btn1");
btn1.addEventListener('click', islem1);


function islem1() {
  alert(1);
}

btn1.addEventListener('click', function() {
  alert(2);
});


// sayfa yüklendiğinde
window.onload = function() {
}
document.addEventListener('DOMContentLoaded', function() {
});
</script>
```

DOM Element Add, Remove, Replace, InsertBefore

DOM yani sayfa içerisine dinamik olarak element ekleyebilir veya eklenen nesneleri kaldırabiliriz.

var fox = document.createElement("img") → Dinamik DOM elemanı oluşturma

parent.appendChild(childEl) → Parent elemanı içerisine childEl elemanını ekler

parent.insertBefore(p1, childEl) → Parent elemanı içerisindeki childEl öncesine p1 elemanını ekler

parent.removeChild(p1) → Parent içerisindeki p1 elemanını kaldırır

https://jsbin.com/rifabek/1/edit?html,js,console,output

```
<div id="icerik"></div>
<div id="icerik2">
    <p id="p1">Paragraf1</p>
</div>
```

```
//var pCem = document.querySelector("#cem");
var pCem = document.createElement("p");
pCem.style.color = '#ff0000';
pCem.style.fontSize = '24px';
pCem.innerHTML = "Paragraf"

//document.body.appendChild(pCem);

var icerik = document.getElementById("icerik");

// pCem elemanını icerik içine ekle
icerik.appendChild(pCem);

var fox = document.createElement("img");
fox.src =
'https://cdn2.iconfinder.com/data/icons/cutecritte
rs/t9foxy_trans.png';
icerik.appendChild(fox);

for (i = 1; i <= 100; i++) {
  //icerik.appendChild(fox);
}
```

DOM Element Add, Remove, Replace, InsertBefore

```
for(i=1; i<=5; i++){
  var fox =
document.createElement("img");
  fox.src =
'https://cdn2.iconfinder.com/data/icon
s/japan-flat-
2/340/fox_animal_wild_nature_wildlife_
cute_tail-256.png'
  fox.style.height = '48px';
  icerik.appendChild(fox);
}

//document.write('Merhaba');
//icerik.innerText += 'Merhaba2';
icerik.innerHTML += 'Merhaba2';
```

```
// icerik2 elemanını seç
let icerik2 = document.getElementById("icerik2");

// dinamik bir başlık oluştur
let hTitle = document.createElement("h1");
hTitle.innerHTML = "Başlık";

// p1 elemanını seç
let p1 = document.getElementById("p1");

// icerik2 kutusu içerisindeki p1 elemanından önce
hTitle elemanını ekle
icerik2.insertBefore(hTitle, p1);

// p1 elemanını HTML den kaldır
//p1.remove();
//icerik2.removeChild(p1);

// icerik2 kutusu içerisindeki p1 elemanını fox
ile değiştir
icerik2.replaceChild(fox, p1);
```

Ödev

Bir şifre alanına girilen şifrenin kolay - orta - zor bir şifre durumunu gösteren uygulama

Bir şifre alanına girilen şifrenin kolay - orta - zor bir şifre durumunu gösteren uygulama.

Kurallar:

Bir büyük harf varsa 3 puan

Bir küçük harf varsa 1 puan

Sayısal değer varsa 1 puan

Özel karakter varsa 3 puan

0-2 arası Kolay, 2-5 Orta, 5> Zor