

# **İstanbul Belediyesi React Eğitimi**

Ders - 4 JavaScript



# Genel Müfredat

Ders Sırası	Tarih	Konu	Süre [Saat]	Ana Eğitmen	Durum	Kayıt	Ders İçeriği	Sınav - Form	Ödev
1	2022-07-18	Tanışma & Ders İşleniş Şekli & VS Code Tanıtımı & VS Code Kurulumu & Notion Tanıtımı & Notion Kurulumu & Proje Takip & Yoklama & GitHub	3	Emin Kartcı	Tamamlandı	Driveda	Paylaşıldı		
2	2022-07-20	Web Nedir & Client-Server & Front-end Backend & HTML CSS JavaScript & Editörler & GitHub Clone & Pull Push Commit & Tag Sistemine Giriş	3	Emin Kartcı	Tamamlandı	Driveda	Paylaşıldı		
3	2022-07-22	HTML Yapısı Tekrar & CSS Özellikleri Tekrar & Taglar Egzersiz & CSS Bağlama Yöntemleri & Padding ve Margin & Renklere Giriş & JavaScript Temelleri & Google Fonts Ekleme & Hesap Makinesi Giriş	3	Kağan Öztürkoğlu	Tamamlandı	Driveda	Paylaşıldı		
4	2022-07-24	JavaScript Detayları & Değişkenler + typeof & Array & Obje & Fonksiyonlar & Returnler & Dosyalar Arası Veri Gönderme & NodeJS Kurulumu ve Package.json tanıtımı	3	Emin & Kağan					
5	2022-07-26	& Express & İlk Server & HTML Map Ekleme & IFrame & Detaylı Taslak Çalışması & PHP Temelleri & Veri Tabanı Tanıtımı & PHP Back-end Kullanımı & Back-end Detayları	3						

# Önceki Dersin Kaydı



REC ●



Google Drive



## 1 Syntax to write CSS

### Selectors

The element(s) on which the style should be applied

### Property and its value

This is the actual style to be applied to the element(s)

```
selectors {  
  ...property: value;  
}
```

## 2 3 places to write CSS

### (A) Inline styles

```
<element style="property: value;">
```

### (B) In the <style> element

```
<head>  
  ...<style>  
    ... selectors { property: value; }  
  ...</style>  
</head>
```

### (C) In a dedicated file (style.css)

& refer that file via the <link> element

```
<head>  
  ...<link rel="stylesheet"  
    href="style.css" />  
</head>
```

## 3 Selectors and their syntax

### Basic Selectors

elementname  
.classname  
#idname  
[attr=value]  
\*

### Combinators

selectorA + selectorB Adjacent sibling  
selectorA ~ selectorB General sibling  
parent > child Direct child  
parent descendant Descendent

### Pseudo Selectors

:active  
:hover  
:visited  
:focus

## 4 Common CSS properties (by group)

### TEXT:

color  
font  
font-family  
font-size  
font-weight  
letter-spacing  
line-height  
text-align  
text-decoration  
text-indent  
text-transform  
vertical-align

### LIST:

list-style  
list-style-image  
list-style-position  
list-style-type

### BACKGROUND:

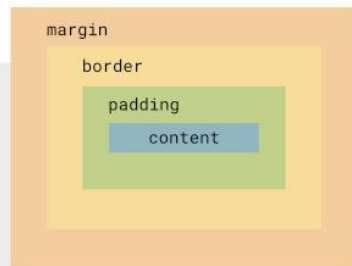
background  
background-attachment  
background-color  
background-image  
background-position  
background-repeat

### DISPLAY:

display  
float  
clear  
overflow  
visibility

### OTHER:

cursor



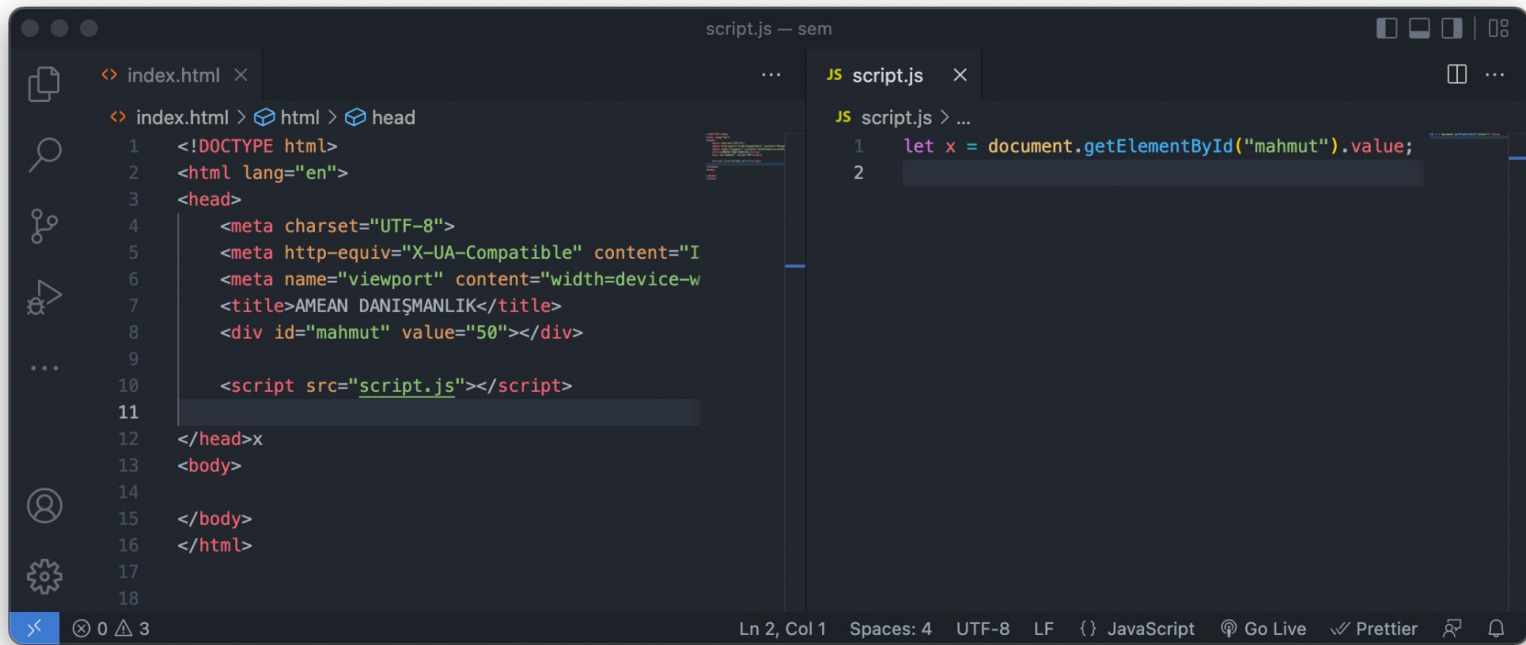
### BOX:

border  
border-color  
border-style  
border-width  
height  
margin  
padding  
width  
box-sizing

### POSITION:

position  
top  
bottom  
left  
right  
z-index

# JS Dosyası Bağlamak



The screenshot shows a code editor with two files open. The left file, `index.html`, contains an HTML document with a `head` section and a `body` section. The `head` section includes a `<script src="script.js"></script>` tag. The `body` section is currently empty. The right file, `script.js`, contains a single line of JavaScript code: `let x = document.getElementById("mahmut").value;`. The editor interface includes a sidebar on the left with icons for file explorer, search, and settings. The bottom status bar shows the current line and column (Ln 2, Col 1), the number of spaces (Spaces: 4), the encoding (UTF-8), the line feed (LF), the language (JavaScript), and the status of Go Live, Prettier, and other extensions.

```
index.html > html > head
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="I
6   <meta name="viewport" content="width=device-w
7   <title>AMEAN DANIŞMANLIK</title>
8   <div id="mahmut" value="50"></div>
9
10  <script src="script.js"></script>
11
12 </head>x
13 <body>
14
15 </body>
16 </html>
17
18

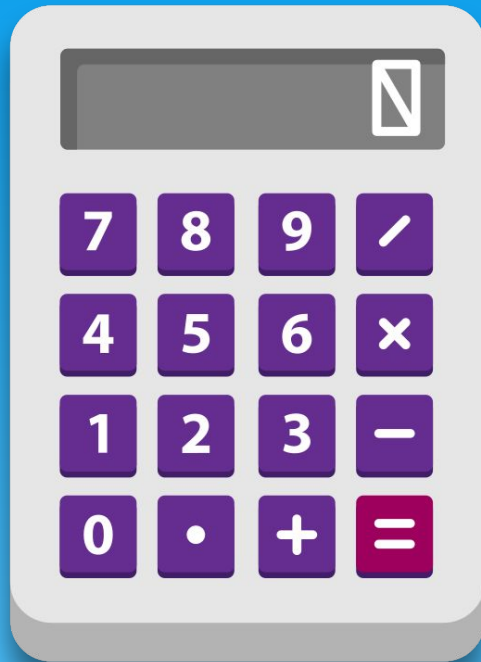
script.js -- sem
JS script.js > ...
1 let x = document.getElementById("mahmut").value;
2
```

# Ödev - 1

Basit bir hesap makinesi yapın.

- Butonlarla sayı girişi olmak zorunda değil. (Yalnızca Tek işlem yapabilen bir makine de olur.)
- Kendi anladıklarınızla yapmaya çalışın.

Ödev Teslim:  
29 Temmuz



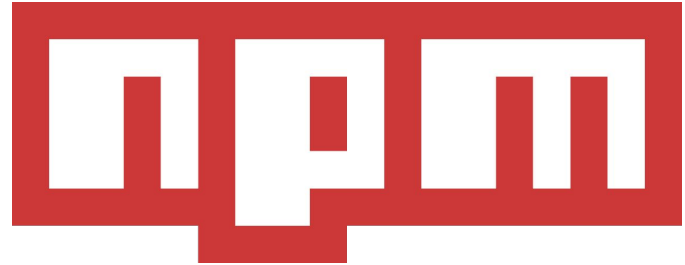
# 1. Oturum

## NodeJS'i Anlamak

# NODE JS ve NPM

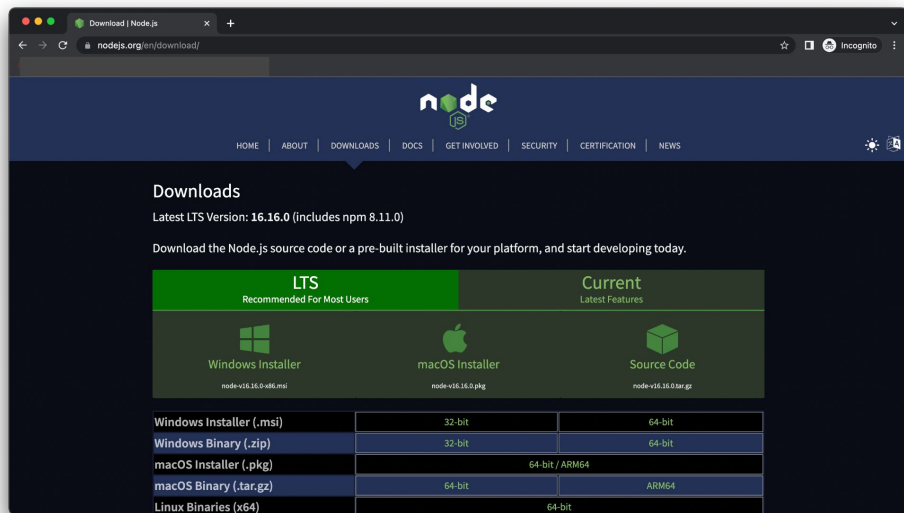
**Node.js** açık kaynaklı programlama dillerinden birisi olup çalışabilmek için herhangi bir sunucuya ihtiyaç duyan uygulamaların tasarlanmasında kullanılır. Dinamik web sayfaları konsepti ile çalışan ve kullanıcıların davranışlarına göre farklı şekillere bürünebilen sistemlerin geliştirilmesinde büyük rol oynar.

**npm**, tarayıcılar ve sunucular gibi online platformlarla etkileşime geçmeye yardımcı olan bir komut satırı aracıdır. Bu araç bir proje gerçekleştirmek için gereken paket yüklemek ve kaldırmakta, sürüm ve bağımlılık yönetiminde yardımcı olur.





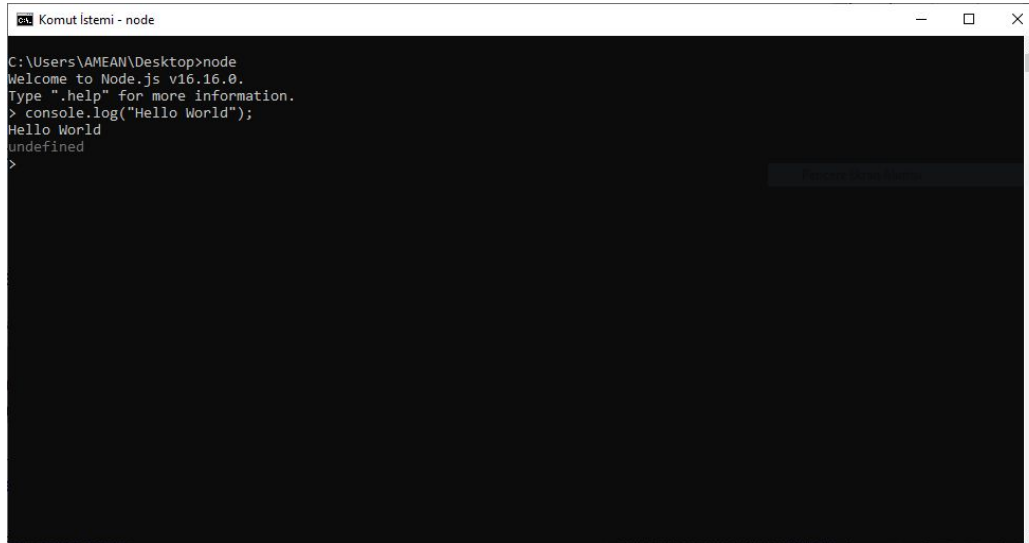
# NodeJS Kurulum



<https://nodejs.org/en/download/>

# NodeJS Terminal

**CMD üzerinden node yazarak nodejs ile javascript kullanmaya başlayabilirsiniz. Değilen tanımlayıp basit işlemleri tarayıcı olmadan çalıştırma imkanınız olur. Server Side JavaScript kavramı tam olarak budur.**



```
Komut İstemi - node
C:\Users\AMEAN\Desktop>node
Welcome to Node.js v16.16.0.
Type ".help" for more information.
> console.log("Hello World");
Hello World
undefined
>
```

# NodeJS Proje Başlangıcı

VS Code üzerinden açılan terminal'de npm init komutunu kullanarak bir node oluşturabilirsiniz. Burada projenize ait genel bilgiler, dependency'ler bulunacaktır.

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL GITLENS

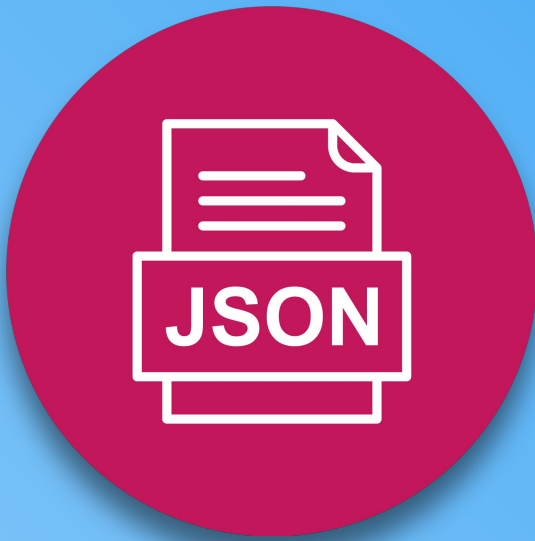
```
PS C:\MAMP\htdocs\IBB_React_Egitimi> npm init
npm WARN config global '--global', '--local' are deprecated. Use '--location=global' instead.
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
```

```
See `npm help init` for definitive documentation on these fields
and exactly what they do.
```

```
Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.
```

```
Press ^C at any time to quit.
```

```
package name: (ibb_react_egitimi) █
```

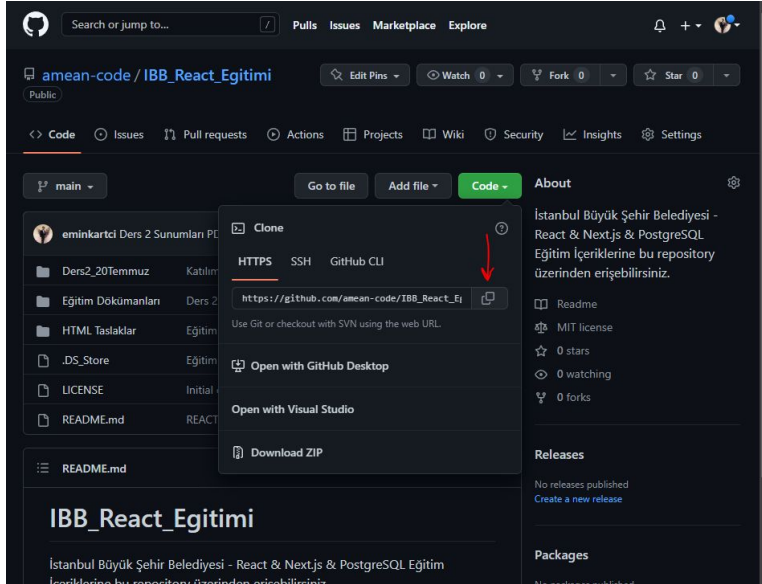


- Bir node projesinin temelinde bulunan bir JSON dosyası. Projeyle ilgili meta verileri tutar ve projenin bağımlılıklarını, komut dosyalarını, sürümünü ve çok daha fazlasını yönetmek için kullanılır.

## Package.JSON Dosyası

# GitHub Proje Baęlamak

Sonradan hazırlanmış bir node'u github repository'sine baęlamak isterseniz git remote add origin [URL] komutunu kullanabilirsiniz.



# 2. Oturum

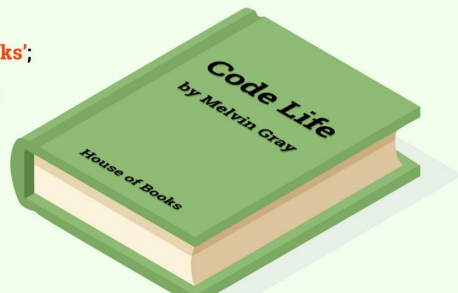
## JavaScripti Tanıyalım

# 1.1 Variables

## Değişkenler

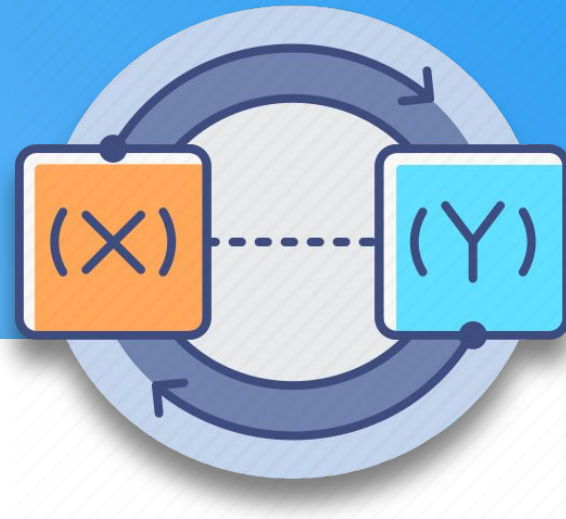
### JavaScript Variables

```
var title = 'Code Life';  
let publisher = 'House of Books';  
const author = 'Melvin Gray';
```



```
<script>  
  var x=1;  
  const y=2;  
  let z=x+y;  
  document.write(z);  
</script>
```

// x ve y değişkeninin değerlerini ekler  
// x ve y'nin toplamını yazdırır



# Yasaklı İsimler

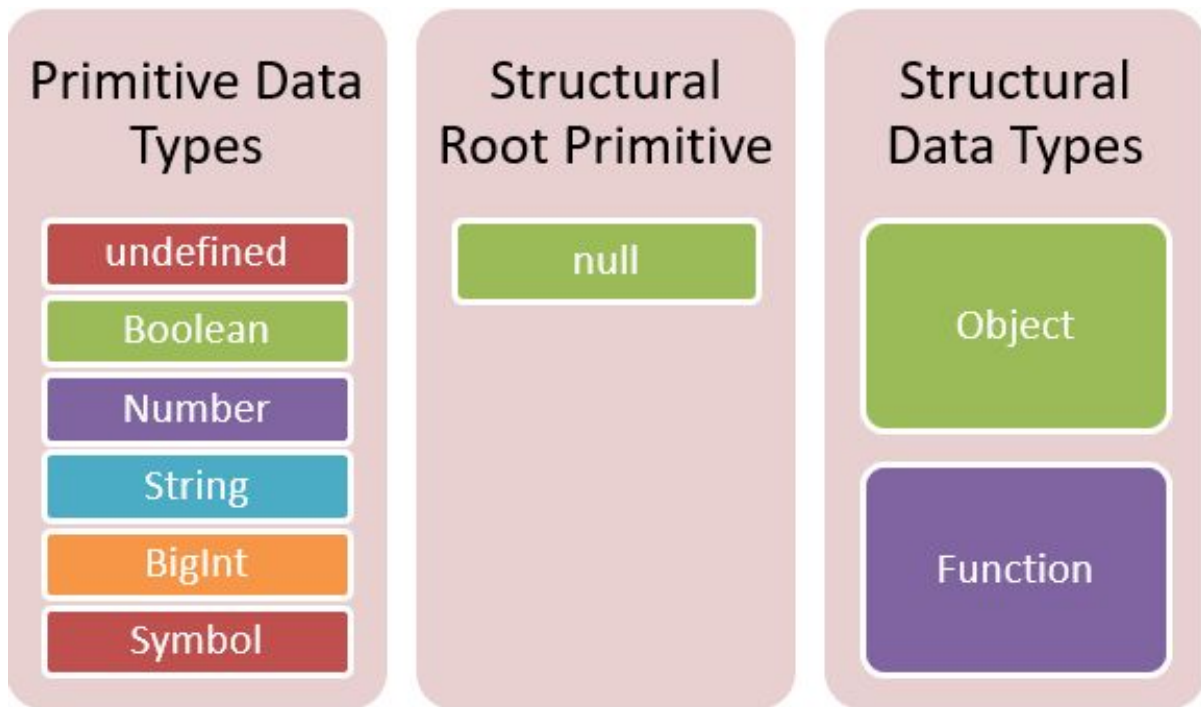
```
var x = 3;  
var y = 12;  
x = x + 2;  
y = x * 3;  
x = 2.5 * y;
```

```
x = ?  
y = ?
```

- Değişken isimlerinde harf, rakam, alt çizgi ve dolar işareti olabilir.
- Değişken isimleri harf, alt çizgi veya dolar işareti ile başlayabilir.
- Değişken isimleri BÜYÜK küçük harf duyarlıdır. (a ve A farklı değişkenlerdir)
- Değişken isimleri arasında boşluk bırakılmaz.
- JavaScript anahtar kelimeleri değişken adı olarak kullanılamaz. (var, debugger, if, while...vb)
- Değişken isimlerinde Türkçe karakter kullanılabilir. Ancak kullanılması tavsiye edilmez.



# JavaScript Değişken Türleri



# Typeof Kullanımı

Typeof Kullanarak konsol üzerinden elinizdeki değişkenlerin tipini görebilirsiniz.

## Dinamik ve Statik Program Dilleri

Javascript dinamik bir dildir yani tanımladığınız değişkenler veri tipleri arasında değişim yapabilir.

```
top
> typeof isApproved
< "boolean"
> typeof firstName
< "undefined"
> typeof selectedColor
< "object"
>
```

```
top
> typeof name
< "string"
> name = 1;
< 1
> typeof name
< "number"
> |
```

# Arrays & Conditionals

## Arrayler ve Kontrol Değişkenleri



# Array

Array data tutan yapı anlamında kullanılabilir. Javascript'te oluşturulan arraylerin belli bir data tipi yoktur yani bir çok çeşitte bilgiyi bir arrayde tutabilirsiniz.



# Array Özellikleri

Sınırsız genişleyebilen torbalardır.

Index: Bilgisayarlar saymaya sıfırdan başlarlar. 3 Eleman olan bir arrayiniz olduğunu varsayalım {"ilk","ikinci","üçüncü"}. "İlk" elemanın indexi 0, "ikinci" elemanın indexi 1, "üçüncü" elemanın indexi ise 2 dir.

- **.length** : arrayin uzunluğu öğrenilebilir
- **[ ]** : sayesinde istenilen indexe ulaşılır
- **.push()** : sonuna eleman ekler
- **.delete()** : indexteki elemanı siler ve yerine undefined bir eleman koyar

## Array Egzersizi

- 3 farklı Değişken oluşturun ve değerleri: 10, Michael ve true olsun.
- Ardından bu değişkenleri bir arraye ekleyin.

# Conditionals

## if, else

```
if (sayi<50) {  
  █  
} else {  
  █  
}
```

```
if (durum) {  
  █  
} else if (durum) {  
  █  
} else {  
  █  
}
```

# Operatör Kullanımı

== eşit değer

=== eşit değer ve aynı tip

!= eşit değil

!== eşit değer değil veya aynı tipte değil

> büyüktür

< küçüktür

>= büyüktür ya da eşittir

<= küçüktür ya da eşittir

&& ve

|| veya

! negatifi

# Functions

Fonksiyonlar programlarımıza  
efektiflik katmak ve düzenli bir  
şekilde ilerlemek için gereklidir.





```
function myFunc() {  
    
}
```

```
function myFunc(params) {  
    
}
```

```
function myFunc(sayi) {  
  console.log(sayi+10);  
}
```

# Fonksiyon Tanımlama

**Parameter (parametre):** Parantezin içindeki değere denir. Program parantezin içine aldığı değeri (sayı) süslü parantezlerin içindeki bölüme yerleştirir.

Parantezin içi dolu olmak zorunda değildir fonksiyon çalıştırıldığında süslü parantezlerin içindeki kodu çalıştırır.

# Fonksiyon Çağırma

Fonksiyonu tanımlamak (önceki slayttaki görseller) çalışması için yeterli değildir. Çalışması için o fonksiyonu çağırmak gerekir.

Argument (argüman):  
Fonksiyonu ÇAĞIRIRKEN içine yazılan değere argüman denir.

Bu durumda 20 sayısı bir argümandır.

```
function benimFonksiyonum() {  
  console.log("Selamlar");  
}
```

```
benimFonksiyonum()
```

```
function myFunc(sayi) {  
  console.log(sayi+10);  
}
```

```
myFunc(20)
```

# OOP - Obje Tabanlı Programlama



## Obje nedir?

**Programlama dilindeki objeler gerçek hayattaki nesnelerin programa aktarılması gibidir. Mesela bir insan objesi oluşturulur ve bu objenin isim yaşı gibi özellikleri olur.**

# Obje Oluşturma

Üstteki gibi süslü parantez içinde oluşturulduğunda obje oluşur.

Nokta Kullanımı: Nokta ile

```
let insan_1 = {  
  isim : "Kağan",  
  yas : 25  
}
```

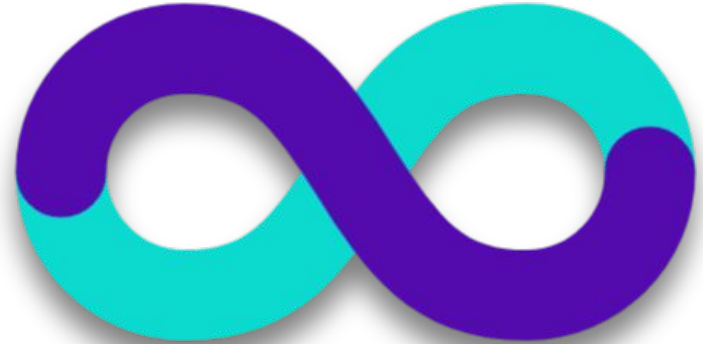
```
> let insan_2 = {  
  isim : "Mahmut",  
  yas : 20  
}  
  
console.log(insan_2.isim);  
Mahmut
```

# While & For

```
while (durum) {  
  console.log("Kod Çalıştı");  
}
```

```
> for (let index = 0; index < 5; index++) {  
  console.log("Kod");  
}
```

5 Kod



# Return

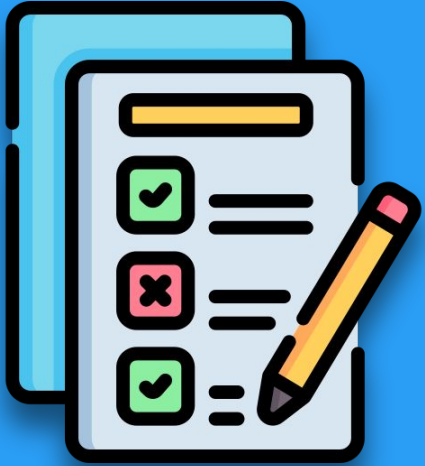


Return'e gelindiğinde, fonksiyon durur ve değer fonksiyonun çağırıldığı yere geri gönderilir.

# 3. Oturum

## JavaScript Egzersizleri

# Egzersiz - 2



Beraber bir not verme fonksiyonu yazalım.  
Konsola `gececekMiyim(64)` yazdığımda  
program çalışmalı.

Parametre olarak girilen sayı;

- 80'den büyükse konsola "Yüksek Başarıyla Geçti.",
- 80 ile 50 arasında ise "Geçti.",
- 50'den düşük ise "Geçemedi.",

yazan bir fonksiyon yazınız.



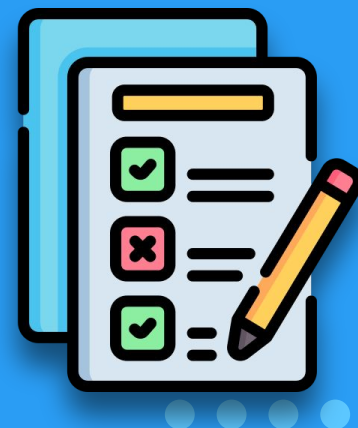
# Egzersiz - 3

Bir çoklu toplama fonksiyonu yazalım.

sayiKadarTopla(10) yazdığımda;

- Birden o sayıya kadar toplama işlemi yapacak.
- $1+2+3+4+5+6+7+8+9= 45$  gibi.

yazan bir fonksiyon yazınız.



# Egzersiz - 4

`2 çarpı 1 = 2`

`2 çarpı 2 = 4`

`2 çarpı 3 = 6`

`2 çarpı 4 = 8`

Bu yazı dizisini konsola yazdıran  
ve bir adet for loop içeren bir  
fonksiyon yazınız.

