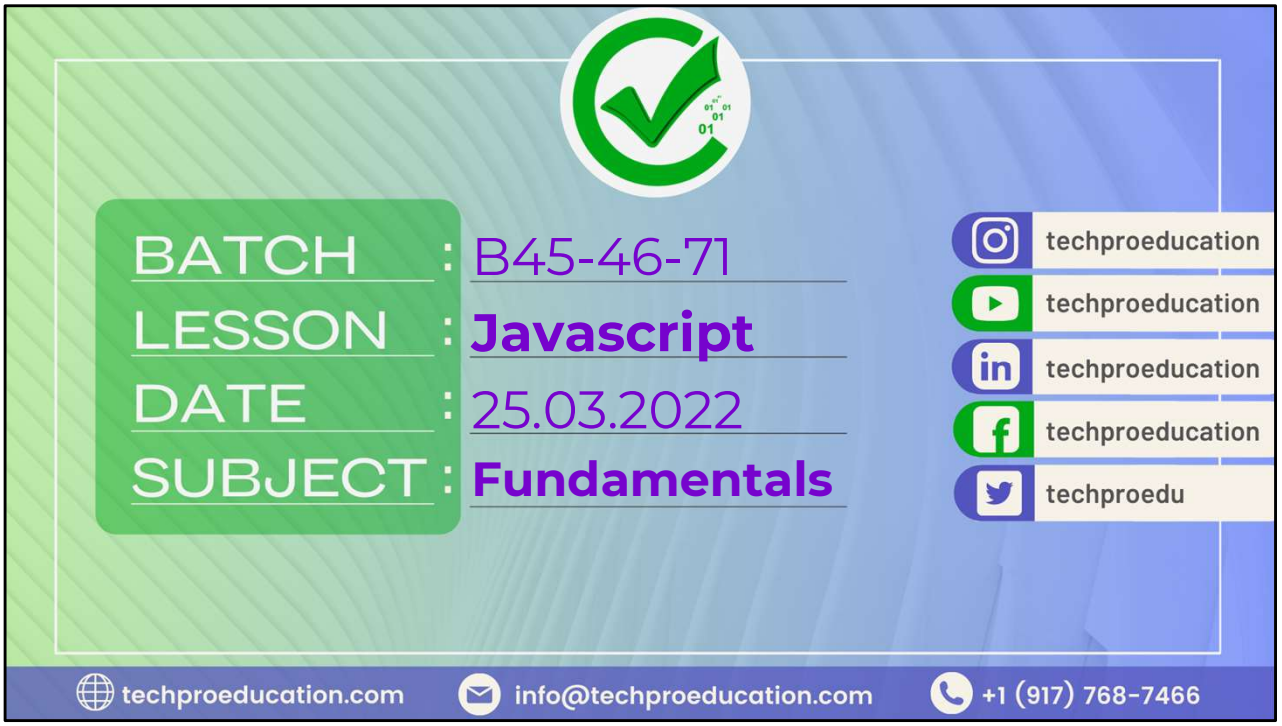
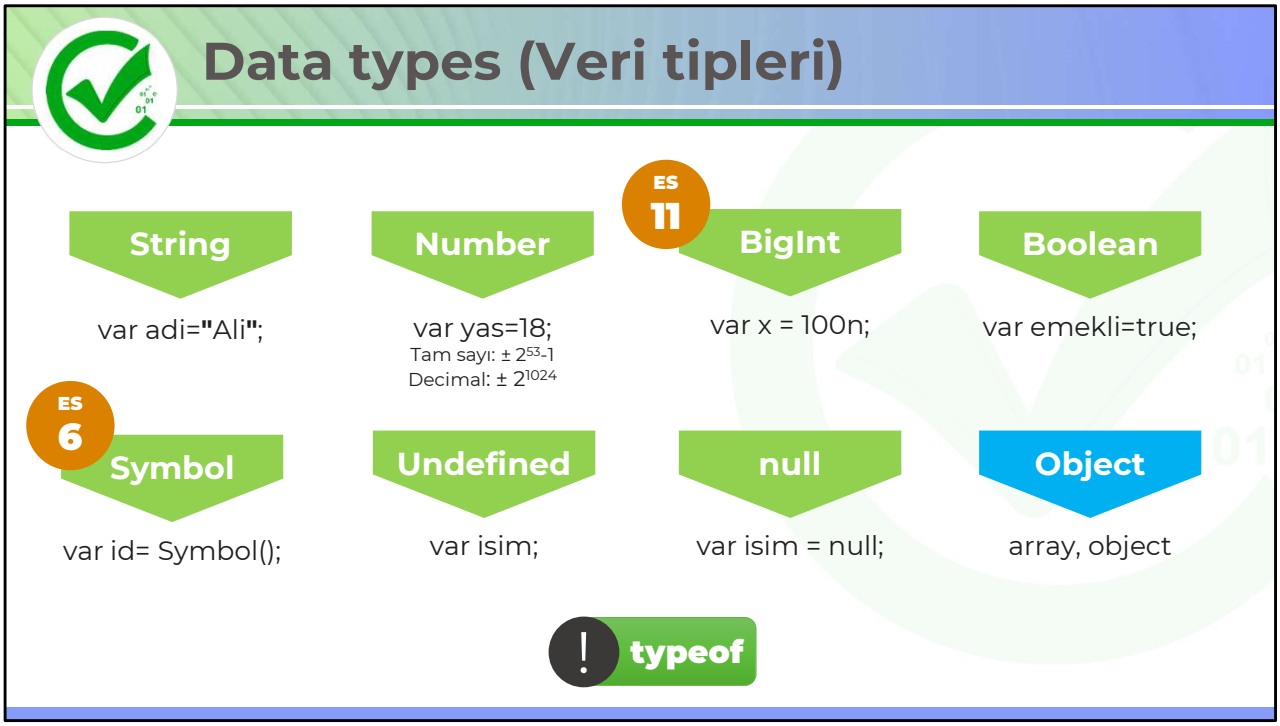

BATCH : B45-46-71
LESSON : Javascript
DATE : 25.03.2022
SUBJECT : Fundamentals
techproeducation
techproeducation
techproeducation
techproeducation
techproedu
techproeducation.com
info@techproeducation.com
+1 (917) 768-7466



Javascript te 6 veri tipi bulunmaktadır. Ancak Javascript te diğer dillerden farklı olarak değişkenler/sabitler tanımlanırken veri türü (data type) belirtilmez. Tanımlama yönteminden Javascript değişkenin/sabitin içinde nasıl bir veri saklayacağını anlar.

1-) Primitive Types (İlkel veri tipleri)

- **String**: metin saklamak için

- **Number**: Sayısal ifade saklamak için. Eğer tamsayı saklanıyorsa ± 2^{53}-1 arası güvenli bir şekilde javascript in işlem yapabileceği aralıktır. Bu aralıktan sonraki sayılar saklanabilir ancak özellikle karşılaştırma işlemlerinde hatalı sonuçlar üretir. Örneğin (Number.MAX_SAFE_INTEGER+2)==(Number.MAX_SAFE_INTEGER+1) sonucu TRUE dönecektir.

- Number.MAX_SAFE_INTEGER ifadesi güvenle işlem yapılabilecek en büyük tam sayı değerini, Number.MIN_SAFE_INTEGER ise en küçüğü verir.

- Ancak ondalıklı sayılar double türüne göre saklanır. Bunda ise sınır yaklaşık ± 2^{1024} aralığıdır.

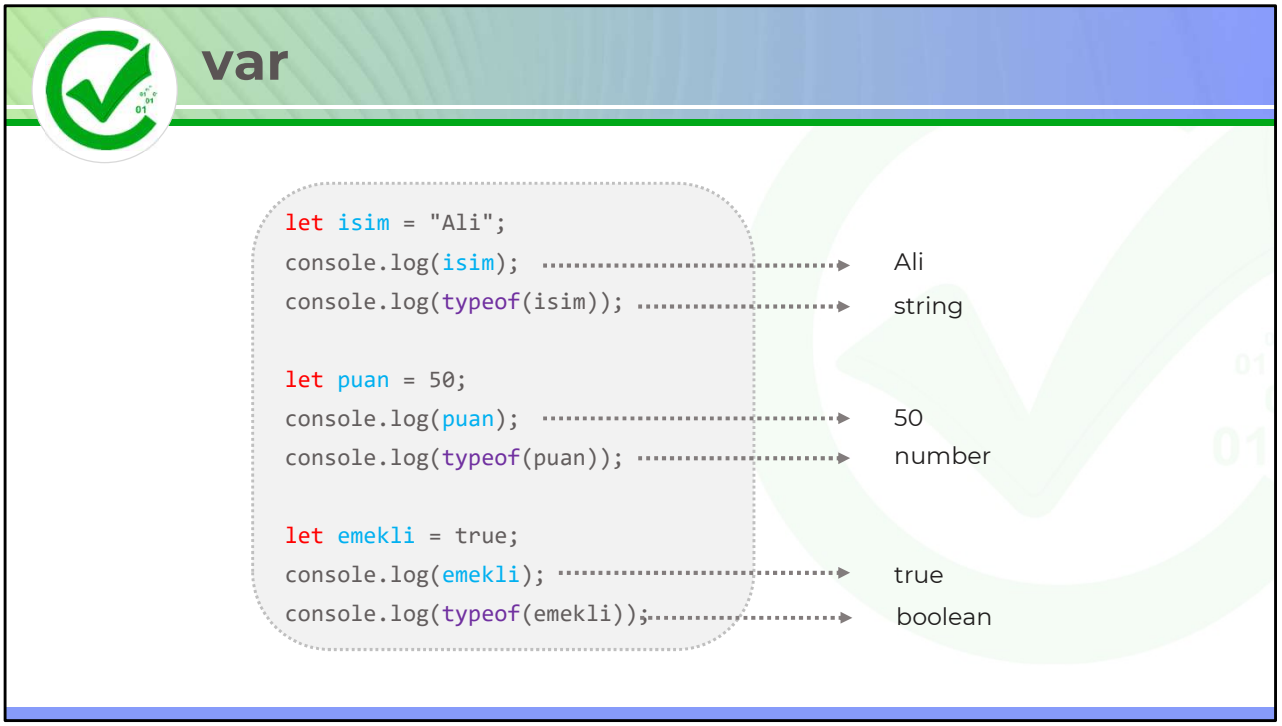
- **BigInt**: ± 2^{53}-1 den daha büyük TAM sayıları saklamak için kullanılır. Javascript in bir sayıyı BgInt olarak algılaması için sayının sonuna «n» karakteri eklenir.

- **Boolean**: True-false değerlerini saklamak için kullanılır.
- **Symbol**: id ve hash gibi benzersiz değerleri güvenle saklamak için kullanılır.
- **Undefined**: Değişlen tanımlanmış ancak değer atanmamış ise Javascript o değişkeni undefined olarak algılar.
- **Null**: Değersiz nesnedir.

2-) Object: new ifadesi ile veya { … }, […] sembolleri ile oluşturulan tüm yapılar nesne olarak kabul edilir. Object türü ilkel veri tipi değildir.

typeof ifadesi ile bir değişkenin değeri öğrenilebilir

```
var x = 15;
console.log(typeof(x));
```


var
let isim = "Ali";
console.log(isim);
Ali
console.log(typeof(isim));
string
let puan = 50;
console.log(puan);
50
console.log(typeof(puan));
number
let emekli = true;
console.log(emekli);
true
console.log(typeof(emekli));
boolean

var

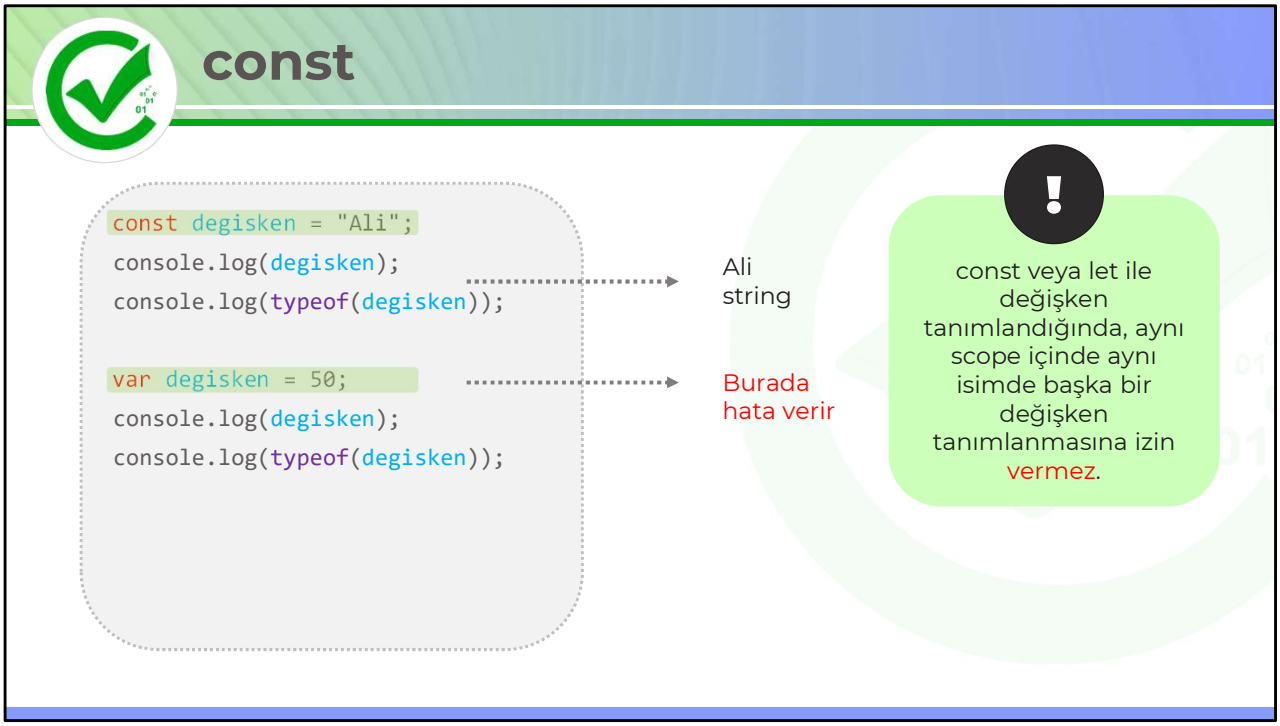
```
var degisken = "Ali";
console.log(degisken);
console.log(typeof(degisken));

var degisken = 50;
console.log(degisken);
console.log(typeof(degisken));

var degisken = true;
console.log(degisken);
console.log(typeof(degisken));
```

Ali
String

50
number

true
boolean

var ile değişken tanımlandığında, aynı isimde başka bir değişken tanımlanmasına izin verir.


const
const degisken = "Ali";
console.log(degisken);
console.log(typeof(degisken));
var degisken = 50;
console.log(degisken);
console.log(typeof(degisken));
Ali
string
Burada
hata verir
const veya let ile değişken tanımlandığında, aynı scope içinde aynı isimde başka bir değişken tanımlanmasına izin vermez.

Operatörler

Aritmetik Operatörler

Karşılaştırma Operatörleri

Mantıksal Operatörler

Atama Operatörler

Aritmetik operatörler

Operatör	Sembol	Örnek
Toplama	+	45+6
Çıkarma	-	35-7
Çarpma	*	25*2
Bölme	/	16/5
Üs Alma	**	5^3
Mod Alma	%	12%5
Bir artırma	++	x++
Bir azaltma	--	y--

Aritmetik operatörler (+)

```
const ekmek = 2;
const yumurta = 30;
const peynir = 40;
const toplamHarcama = ekmek + peynir + yumurta;
console.log("HARCAMA:" + toplamHarcama + " TL");
```

ÇIKTI
HARCAMA: 72 TL

```
const ad = 'Ali';
const soyAd = 'Gel';
console.log(ad + soyAd);
console.log(ad + ' ' + soyAd);
```

ÇIKTI
AliGel
Ali Gel

ÖNEMLİ
+ operatörü ile **string** birleştirme de gerçekleştirilebilir .

```
const x = 5;
const y = "5";
const birlestir = x + y;
console.log(birlestir);
```

ÇIKTI
55

Tip dönüşümleri

```
const para = "100";
console.log(para + 15);
console.log(Number(para) + 15);
```

ÇIKTI
10015

ÇIKTI
115

ÖNEMLİ
Number() fonksiyonu tip çevrimi yapılabilir.

```
const dil = "Javascript";
console.log(Number(dil));
console.log(Number("123abc"));
```

ÇIKTI
NaN

ÇIKTI
NaN

ÖNEMLİ
Number() fonksiyonu harfleri sayıya çevirmeyeceği için NaN (Not a number – Sayı değil) döndürüyor.

```
const para = 5400;
console.log(para+15);
console.log(String(para)+15);
```

ÇIKTI
5415

ÇIKTI
540015

ÖNEMLİ
String() fonksiyonu ile verilen İfadeyi string'e çevirmek mümkündür.

Tip dönüşümleri

```
const s1 = 5;
const s2 = -7;
const isim = "John";
console.log(Boolean(isim));
console.log(Boolean(s1));
console.log(Boolean(s2));
```

ÇIKTI
true
true
true

ÖNEMLİ
0, **null**, **undefined**, **NaN**, ve " " Javascript tarafından **false** olarak kabul edilir.

Diğer değerler Boolean'a çevrildiğinde **true** olarak kabul edilir.

```
const sifir = 0, nal = null;
const tanimsiz = undefined;
const bos = "", sayiDegil = NaN;

console.log(Boolean(sıfır),Boolean(nal));
console.log(Boolean(tanımsız), Boolean(boş));
console.log(Boolean(sayıDeğil));
```

ÇIKTI
false
false
false

İşlem önceliği

Aritmetik işlemler yapılırken bilgisayar operatör önceliğine göre işlem yapar.

! Öncelik seviyesi aynıysa soldaki ifadeyi önce yapar.

( ) Parantez	
** Üs alma	
* Çarpma	/ Bölme
+ Toplama	- Çıkarma

İşlem önceliği

8/2*(2+2)	16
30 – 3**2 / 3 + 10	37
16 / 2 * 3 – 2**(4 / 2)	20
(14 * 2 / 7)**2 / 4 + 5	9

String literals

ES 6

Metinleri daha dinamik bir şekilde birleştirmek için **string şablonları** (string literals, template literals) kullanabiliriz. Bu birleştirme işlemine de **string interpolation** denir.

```
let isim = "Ali";
let soyisim = 'Gel';
```

Tek tırnak veya çit tırnak

```
let isim = "Ali";
Console.log(`Merhaba ${isim}`);
```

string interpolation bactick ile yapılır

Backtick multiline text tanımlamaya da izin verir.

Aritmetik operatörler (-)

```
const yil = 2021;
const dogumTarihi = 1980;
const yas = yil - dogumTarihi;
console.log("YAŞ:" + yas);
console.log("YAŞ:" + yil - dogumTarihi);
```

ÇIKTI

YAŞ: 41

NaN

ÖNEMLİ

Ekstra **parantez** kullanılmaz ise **string** birleştirme yapmaya çalışır. – den dolayı birleştiremez ve **NaN** döndürür.

NaN = Not a Number (Sayı değil)

Aritmetik operatörler (*, **)

```
const pi = 3;
const r = 3;
const alan = pi*r**2;
const çevre = 2*pi*r
console.log(çevre, alan);
console.log("ÇEVRE:" + çevre, "ALAN:"+ alan);
```

ÇIKTI

18 27

ÇEVRE:18 ALAN:27

Aritmetik operatörler (++, --, %)

```
let a = 3;  let b = ++a;  let c = --a;
console.log(a,b,c);

a += 5;
console.log(a);
```

ÇIKTI
3 4 3

ÇIKTI
8

```
const z = 3;
let k = z++;
console.log(k);
```

HATA
const değişkenin değeri arttırılamaz.

```
const sayi = 123;
console.log("Birler Basamağı:" + sayi%10);
```

ÇIKTI
Birler Basamağı: 3