JavaScript

1. Hyrje ne JavaScript
   1. Cfare eshte JavaScript
   2. Krijimi i nje mjedisi lokal per zhvillim dhe tesitm
   3. Console
   4. Komentet
   5. Tipet e te dhenave
   6. Operatoret Aritmetike
   7. Properties
   8. Metodat
   9. Objektet built-in
   10. Variablat
   11. Krijimi i variablave nepermjet var, let dhe const. Ndryshimet midis tyre
   12. Operatoret aritmetike te caktimit
   13. Operatoret e inkrementimit dhe dekrementimit
   14. Bashkimi i string-eve (string concatenation) dhe string interpolation
   15. Operatori typeof
   16. **Cfare eshte JavaScript**

JavaScript eshte nje gjuhe skriptimi ose programimi e cila mundeson zhvillimin e aplikacioneve web. JavaScript eshte i njohur krysisht si gjuha e shumices se shfletuesve moderne te internetit dhe mundeson sjelljen dinamike dhe nderveprimin e perdoruesit ne shumicen e aplikacioneve online.

JavaScript eshte nje prej tre shtresave te teknologjive standarde te internetit, dy prej te cilave jane HTML dhe CSS.

HTML sherben per te definuar dhe strukturuar permbajtjen e nje aplikacioni online, si per shembull titujt, paragrafet, listat, tabelat, imazhet dhe elemente te tjere.

CSS na sherben per te stiluar permbajtjen dhe strukturen e definuar ne HTML. Me ane te saj mund te manipulojme ngjyrat, madhesine e shkrimit, vendosjen ne kolona te permbajtjes dhe shume karakteristika te tjera.

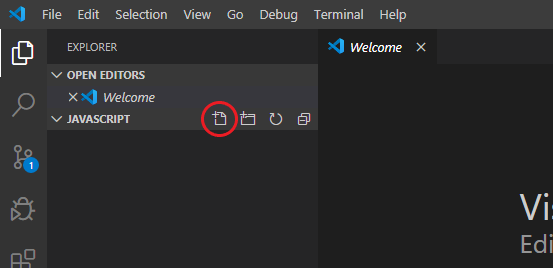
* 1. **Krijimi i nje mjedisi lokal per zhvillim dhe testim**

Per te koduar ne JavaScript duhet te zgjedhim nje code editor. Ky liber rekomandon perdorimin e Visual Studio Code, i cili mund te shkarkohet te linku me poshte:

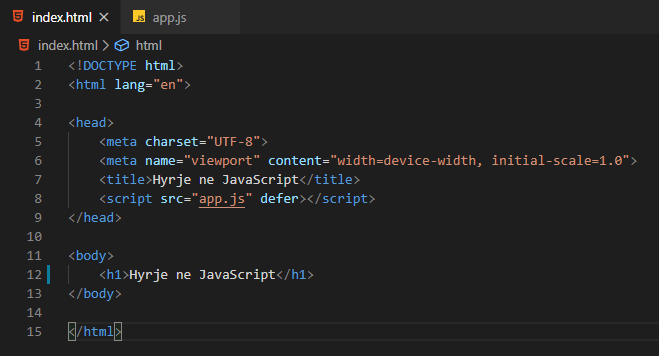
<https://code.visualstudio.com/>

Me pas do te ndertojme strukturen baze te aplikacionit tone, duke ndjekur hapat me poshte.

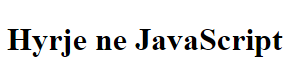
1. Krijoni nje direktori te re (emertojeni sipas deshires).
2. Pasi te keni shkarkuar dhe instaluar editorin e preferuar, hapeni ate dhe navigoni te   
   File > Open Folder
3. Seleksiononi direktorine e krijuar ne hapin e pare
4. Aplikacioni do te permbaje dy file, perkatesish: index.html, dhe app.js. File-et e reja mund te krijohen duke klikuar te butoni New File



1. File html do te permbaje kodin me poshte:



1. Per tu siguruar qe mjedisi i testimit eshte funksional, mjafton te ekzekutoni aplikacionin duke naviguar te direktoria e krijuar dhe double-click te file index.html.  
   Rezultati do jete faqja e meposhtme:



**1.3 Console**

Console eshte nje panel qe shfaq mesazhe te rendesishme per zhvilluesit. Ne JavaScript fjala kyce console i referohet nje objekti, nje koleksioni te dhenash dhe veprimesh, te cilat mund t’i perdorim ne kodin tone. Fjalet kyce jane fjale te cilat jane pjese te gjuhes JavaScript, keshtu qe kompjuteri do t’i njohe dhe do t’i trajtoje ato specifikisht.

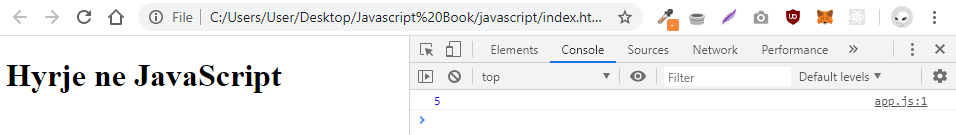
Nje veprim, ose metode, qe eshte pjese e objektit console eshte metoda .log(). Kur shkruajme console.log() ato qe vendosim brenda kllapave do te printohen ne console.

Do te jete shume e dobishme per ne te printojme vlera ne console, keshtu qe mund te shohim punen qe po bejme.

Te file app.js shkruani kodin me poshte

console.log(5);

Per te aksesuar console ne shfletuesin tone mjafton te klikojme me te djathen te faqja e aplikacionit tone dhe te klikojme alternativen e fundit: Inspect; ose nepermjet tastieres me komanden Ctrl + Shift + I. 

Me pas navigoni te opsioni Console  
  


Ky shembull printon numrin 5 ne console. Pikepresja ne fund te kodit tone tregon fundin e rreshtit, ose fundin e deklarimit. Me vone do te shohim se ne mund te perdorim console.log() per te printuar lloje te ndryshme te dhenash.

**1.4 Komentet**

Shpesh here mund te ndodhe qe kodi yne pas pak kohesh te behet i veshtire per tu kuptuar. Gjithashtu, me kodin tone mund te punoje dhe nje zhvillues tjeter. Per keto arsye, eshte shpesh e dobishme te vendosen komente ne kod, per zhvilluesit e tjere ose per veten tone.

Komentet ne JavaScript mund te perdoren per te shpjeguar se cfare po ben kodi, te lene udhezime per zhvilluesit e tjere, apo per ta bere ate me te lexueshem.

Komentet ne kodin tone do te injorohen nga kopjuteri, nderkohe qe pjesa tjeter e programit do te ekzekutohet. Keto komente ekzistojne vetem per lexuesit njerezore.

Ekzistojne dy menyra per te komentuar kodin ne JavaScript:

1. Komenti i nje linje te vetme (single line comment)

Do te komentoje nje rresht te vetem dhe shenohet me // ne fillim te rreshtit, p.sh:  
  
//printon 5 ne console

console.log(5);

Gjithashtu, komenti i nje linje te vetme mund te behet edhe ne fund te nje deklarimi:

console.log(5); // printon numrin 5

1. Komenti i linjave te shumfishta (multi-line comment)

Nje koment multi-line do te komentoje disa rreshta dhe do te shenohet me /\* per te filluar komentin, dhe \*/ per ta perfunduar ate.

/\*

Gjithcka brenda ketij blloku eshte e komentuar

console.log(10);

Asnje nga rreshtat nuk do te ekzekutohet

console.log(99);

\*/

Kjo sintakse gjithashtu mund te perdoret per te komentuar dicka ne mes te nje linje kodesh:

console.log(/\* kjo pjese nuk do te ekzekutohet \*/ 42); // printon numrin 42

**1.5 Tipet e te dhenave**

Te gjitha gjuhet e programimit kane lloje te dhenash, megjithese ato ndryshojne ne varesi te gjuhes. Llojet e te dhenave jane klasifikimet qe ne u japim llojeve te ndryshme te te dhenave qe perodorim ne programim. Standarti i fundit I ECMAScript percakton tete lloje te te dhenave:

* Number
* String
* Boolean
* Null
* Undefined
* Symbol
* BigInt
* Object

7 llojet e para konsiderohen si llojet e te dhenave primitive. Ato jane llojet me themelore te te dhenave ne gjuhe. Objektet jane me komplekse. Ne shkencat kompjuterike, nje objekt eshte nje vlere ne kujtese e cila mund te referohet nga nje identifikues.

1. *Number*

C’do numer, perfshire numrat me presje: 4, 8, 42, 28.28.

console.log(42);

1. String

C’do grupin i karaktereve ne tastiere (shkronja, numra, hapesira, simbole, etj) i rrethuar nga thonjeza teke: ‘…’ ose te dyfishta “…”.

console.log(‘Ky eshte nje varg karakteresh, tipi i te cilit eshte string.’)

1. Boolean

Ky lloj i te dhenave ka vetem dy vlera te mundshme: true ose false (pa thonjeza). Mendojeni boolean si aktivizues dhe caktivisues ose si pergjigje per nje pyetje “po” ose “jo”.

1. Null

Ky lloj i te dhenave perfaqeson mungesen e qellimshme te nje vlere, dhe shenohet me fjalen kyce null (pa thonjeza).

1. Undefined

Ky lloj I te dhenave shenohet me fjalen kyce undefined (pa thonjeza). Ai gjithashtu perfaqeson mungesen e nje vlere por ka nje perdorim te ndryshem nga null.

1. Symbol

Simbolet jane identifikues unik dhe te pandryshueshem te cilet mund te perdoren si key per nje property ne nje objekt. Do ta flasim per properties ne nje prej temave ne vijim.

1. BigInt

Lloji BigInt mund te perfaqesoje nje numer te plote me precizion arbitrar. Nje BigInt mund te krijohet duke bashkangjitur n ne fund te nje numri te plote ose duke thirrur konstruktoirin (konstruktori shpjegohet ne kapitujt ne vijim).

1. Object

Koleksion te dhenash te cilat kane lidhje me njera-tjetren.

**1.6 Operatoret Aritmetike**

Operatoret aritmetike performojne veprime matematikore me numrat dhe kthejne nje vlere te vetme numerike. JavaScript ka disa operatore te integruar aritmetike, disa prej te cilave jane:

1. Shtimi: +

Operatori i shtimit prodhon shumen e vlerave numerike:

console.log(3 + 4); // printon 7

Ne rast se operatori i shtimit perdoret ne dy string (vargje karakteresh), atehere ai shton vargun e djathte ne vargun e majte:

console.log(‘Pershendetje ’ + ‘bote’); // printon Pershendetje bote ne console.

Procesi i shtimit te nje string me nje tjeter quhet string concatenation (bashkim i vargjeve). Vini re se si ne shembullin me lart, eshte dashur te perfshijme nje hapesire ne fund te vargut te pare. Kompjuteri do te bashkoje ekzaktesisht dy vargjet, keshtu qe na duhet te sigurohemi qe te perfshijme hapesiren qe donim midis dy vargjeve. Ne munge te saj, rezultati do te ishte: Pershendetjebote.

1. Zbritja: -

Operatori i zbritjes prodhon diferencen e vlerave numerike:

console.log(7 - 4); // printon 3

1. Shumezimi: \*

Operatori i shumezimit prodhon produktin e vlerave numerike:

console.log(4 \* 3); // printon 12

1. Pjestimi: /

Operatori i pjestimit prodhon rezultatin e shprehjes, ku vlera e majte eshte dividenti dhe vlera e djathe eshte pjestuesi:

console.log(9 / 3) // printon 3

1. Mbetja: %

Operatori i mbetjes kthen mbetjen e pjestimit:

console.log(11 % 3); // printon 2

11 % 3 jep si rezultat numrin 2 sepse 3 hyn 3 here te 11, duke lene mbejten 2.

1. Ngritja ne fuqi: \*\*

Operatori i ngritjes ne fuqi kthen rezultatin e ngritjes ne fuqi te vleres numerike ne anen e majte me ekosponentin

console.log(2 \*\* 2) // printon 4

**1.7 Properties**

Kur prezantoni nje pjese te re te te dhenave ne nje program JavaScript, shfletuesi e ruan ate si nje instance te llojit te te dhenes. Cdo instance string ka nje property te quajtur length e cila ruan numrin e karaktereve ne ate string (varg karakteresh). Ky informacion mund te merret duke bashkengjitur string-un me nje pike “.” dhe emrin e property:

console.log(‘Pershendetje’.length); // printon 12

. eshte nje tjeret operator i cili quhet dot operator.

Ne shembullin e mesiperm, vlera e ruajtur ne property-n length (gjatesi), merret nga instanca e string-ut ‘Pershendetje’. Programi printon 12 ne console, sepse Pershendetje ka 12 karaktere.

**1.8 Metodata**

Metodat jane veprime qe mund te kryejme me nje instance te nje te dhene.

Ne i ekzekutojme, ose perdorim keto metoda, duke shtuar nje pike (operatori dot) ne fund te instances, emrin e metodes, dhe kllapat e hapjes dhe te mbylljes.

Kete sintakse e kemi hasur dhe me pare, kur kemi perodrur console.log(). Kur perdorim console.log() ne ekzekutojme metoden .log() ne objektin console.

JavaScript ofron disa metoda per vargjet e karaktereve. Disa shembuj:

console.log(‘pershendetje’.toUpperCase()); // printon PERSHENDETJE

console.log(‘Pershendetje’.startsWith(‘P’)); // printon true

Le te shohim secilin nga shembujt me ne detaj.

Ne rreshtin e pare, metoda .toUpperCase() thirret ne instancen e stringut ‘pershendetje’. Rezultati printohet ne console. Kjo metode kthen nje string me te gjitha shkronjat e medha: ‘PERSHENDETJE’.

Ne rreshtin e dyte, metoda .startsWith() thirret ne instancen e stringut ‘Pershendetje’. Kjo metode gjithashtu pranon karakterin “H” si nje input, ose argument, midis kllapave. Meqenese vargu “Pershendetje” fillon me shkronjen “P”, metoda kthen boolean-in true.

Zhvilluesit perdorin dokumentacionin si nje mjet referimi. Ai pershkruan fjalet kyce, metodat dhe sintaksen e JavaScript. Dokumentacionin mund ta gjeni te linku me poshte:  
<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String>

**1.9 Objektet built-in**

Pervec objektit console, ka dhe objekte te tjera te integruara ne JavaScript. Keto objekte te integruara jane plot me funksionalitete te dobishme.

Per shembull, nese keni dashur te kryeni veprime aritmetike me komplekse, JavaScript ka objektin e integruar Math.

Ashtu si e kemi permendur ne seksionet pararendese, objektet kane metoda. Nje prej metodave te objektit Math eshte metoda .random():

console.log(Math.random()); // printon nje numer rastesor midis 0 dhe 1

Ne shembullin e mesiperm, kemi thirrur metoden .random() duke bashkengjitur emrin e objektit me operatorin dot (pike), emrin e metodes dhe kllapat e hapura dhe te mbyllura. Kjo metode kthen nje numer te rastit midis 0 dhe 1.

Per te gjeneruar nje numer rastesor midis 0 dhe 50, ne mund te shumezojme kete rezultat me 50:

Math.random() \* 50;

Shembulli me lart mund te ktheje nje numer me presje dhjetore. Per tu siguruar qe pergjigja eshte nje numer i plote, ne mund te perfitojme nga nje metode tjeter e dobishme nga objekti Math, e quajtur Math.floor().

Math.floor() pranon nje numer me presje dhjetore si argument dhe e rrumbullakos ate ne numrin e plote me te afert. Math.floor() mund te perdoret per te rrumbullakosur nje numer rastesor si me poshte:

Math.floor(Math.random() \* 50);

Ne kete rast:

Math.random() krijon nje numer te rastit midis 0 dhe 1.  
Me pas ne e shumezojme ate numer me 50, keshtu qe tani kemi nje numer midis 0 dhe 50.  
Ne hapin tjeter, Math.floor() rrumbullakos numrin ne numrin e plote me te afert.

Per me shume informacion ne lidhje me properties dhe metodat e objektit Math, ndiqni dokumentacionin te linku me poshte:

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math>

**1.10 Variablat**

Ne programim, variablat jane kontejnere per te ruajtur vlerat e te dhenave, informacioni i te cilave jeton ne kujtesen e kompjuterit. Informacioni i ruajtur ne variabla, si p.sh nje emer perdoruesi, numer llogarie, apo edhe pershendetjet e personalizuara mund te gjenden me pas ne memorie.

Variablat gjithashtu sigurojne nje menyre per etiketimin e te dhenave me nje emer pershkrues, keshtu qe programi mund te kuptohet me qarte nga nje programues tjeter dhe nga ne.

Me pak fjale, variablat etiketojne dhe ruajne te dhenat ne memorie. Ekzistojne vetem disa gjera qe mund te beni me variablat:

Krijimi i nje variabli me nje emer pershkrues.  
Ruani ose perditesoni informacionin e ruajtur ne nje variable.  
Ti referoheni apo te “merrni” informacionin e ruajtur ne nje variabel.

Eshte e rendesishme te behet dallimi midis variablave dhe vlerave, variablat nuk jane vlera; ato permbajne vlera dhe i perfaqesojne ato me nje emer.

**1.11 Krijimi i variablave nepermjet var, let dhe const. Ndryshimet midis tyre**

Krijimi i nje variabli nepermjet fjales kyce var

Versioni ES6 i JavaScript ne vitin 2015 solli shume ndryshime. Nje nga ndryshimet me te medha ishte prezanitmi i dy keywords te reja, let dhe const, per te krijuar ose deklaruar variabla. Para ES6, programuesit mund te perdornin vetem fjalen kyce var per te deklaruar variabla.

var emriIm = ‘John’;  
console.log(emriIm); // printon John ne console

Le ta shqyrtojme shembullin e mesiperm:

1. var: shkurtimi per ndryshore, eshte nje fjale kyce ne JavaScript qe krijon, ose deklaron nje variabel te ri.
2. emriIm eshte emri i ndryshores. Shkrimi i fjales te dyte me shkronje kapitale eshte nje standart qe perdoret ne JavaScript i quajtur camel case. Nepermjet ketij standarti te gjitha fjalet grupohen ne nje, fjala e pare fillon me shkronje te vogel, dhe cdo fjale pasardhese fillon me shkronje te madhe. (p.sh camelCaseCdoFjale).
3. = eshte operatori i caktimit. Operatori cakton vleren ‘John’ ne variablin emriIm.
4. John eshte vlera e caktuar te variabli emriIm. Mund te themi gjithashtu se variabli emriIm eshte inicializuar me nje vlere John.
5. Pasi deklarohet variabli, string-u John printohet ne console duke ju referuar emrit te variablit:  
   console.log(emriIm);

Ekzistojne disa rregulla te pergjithshme per emertimin e variablave:

* Emrat e variablave nuk mund te fillojne me numra.
* Emrat e variablave jane case sensitive, keshtu qe ne rast se do te kishim dy variabla, p.sh, emri dhe Emri, ato do te ishin dy variabla te ndryshme. Konsiderohet praktike e keqe krijimi i dy variablave me te njejtin emer duke perdorur madhesi karkteresh te ndryshme.
* Emrat e variablave nuk mund te jene te njejte me fjalet kyce

Krijimi i variablave me fjalen kyce let

Fjala kyce let u prezantua ne versionin ES6. Kjo fjale kyce sinjalizon se variabli mund te ri-caktohet me nje vlere tjeter. Shikoni shembullin:

let dita = ‘E shtune’;  
console.log(dita); // rezultati: E shtune  
dita = ‘E diele’;  
console.log(dita); // rezultati: E diele

Nje concept tjeter per te cilin duhet te jemi te vetedijshem kur perdorim let (apo dhe var) eshte qe ne mund te deklarojme nje variable pa i caktuar nje vlere variablit. Ne kete rast, variabli do te inicializohet automatikisht me nje vlere udefined.

let cmimi;  
console.log(cmimi); // rezultati: undefined  
cmimi = 350;  
console.log(cmimi); // rezultati: 350

Vini re ne shembullin e mesiperm:  
Nese nuk i caktojme nje vlere nje variabli te deklaruar me fjalen kyce let, let automatikisht ka nje vlere undefined.  
Gjithashut, ne mund te ri-caktojme vleren e variablit.

Krijimi i variablave me fjalen kyce const

Ashtu si fjala kyce let, const u prezantua gjithashtu ne ES6, dhe eshte shkurtim per fjalen konstante. Njelloj si me var dhe let, const mundeson ruajtjen e vlerave ne nje variabel const. Menyra e deklarimit eshte e njejte me var dhe const. Per shembull:

const numriLlogarise = 12345678;  
console.log(numriLlogarise); // rezultati: 12345678

Megjithate, nje variable const nuk mund te ri-caktohet sepse eshte constant. Nese perpiqeni te ri-caktoni nje variable const, do ju kthehet nje gabim TypeError.

Variablave te deklaruar me const, duhet t’ju caktohet nje vlere kur deklarohen. Nese perpiqeni te deklaroni nje variable const pa nje vlere, do ju kthehet nje gabim SyntaxError.

Nese jeni ne dileme per te vendosur se cilen fjale kyce duhet te perdorni, let apo const, mendoni nese do ju duhet te ri-caktoni vleren e variablit me vone. Nese keni nevoje ta ricaktoni, perdorni let, perndryshe, perdorni const.

**1.12 Operatoret aritmetike te caktimit**

Le te shqytrojme se si mund t’i perdorim variablat dhe operatoret aritmetike per te llogaritur vlerat e reja dhe t’i caktojme ato ne nje variable. Shikoni shembullin me poshte:

let x = 4;  
x = x + 1;  
console.log(x); // printon 5

Ne shembullin e mesiperm, kemi krijuar variablin x me numrin 4 te caktuar per te. Rreshti i meposhtem, x = x + 1, rrit vleren e x nga 4 ne 5.

Nje menyre tjeter per te ri-caktuar x-in pas kryerjes se disa veprimeve matematikore me te, eshte perdorimi i operatoreve te integruar te caktimit matematikor. Ne mund te rishkruajme kodin e mesiperm keshtu:

let x = 4;  
x += 1;  
console.log(x); // printon 5

Ne shembullin e dyte, ne kemi perdorur operatorin e caktimit += per te ri-caktuar x. Fillimisht kryhet veprimi matematikor i opratorit te pare + duke perdorur numrin ne te djathe, pastaj ri-caktojme x ne vleren e llogaritur.

Operatore te tjere aritmetik te caktimit jane:

* -=
* \*=
* /=

Keta operatore funksionojne ne menyre te ngjashme me +=.

let a = 20;  
a -= 5 // mund te shkruhet si x = x - 5  
console.log(a); // rezultati: 15

let b = 50;  
b \*= 2; // mund te shkruhet si b = b \*2  
console.log(b); // rezultati: 100

let c = 8;  
c /= 2; // Mund te shkruhet si c = c / 2  
console.log(c); // rezultati: 4

**1.13 Operatoret e inkrementimit dhe dekrementimit**

Operatori i inkrementimit do te rrise vleren e variablit me 1, nderkohe qe operatori i dekrementimit do ta ule ate me 1. Per shembull:

let a = 10;  
a++;  
console.log(a); // rezultati: 11

let b = 20;  
b--;  
console.log(b); // rezultati: 19

Ashtu si operatoret e meparshem te caktimit matematikor (+=, -=, \*=, /=), vlera e variables perditesohet dhe caktohet si vlere e re e saj.

**1.14 Bashkimi i string-eve (string concatenation) dhe string interpolation**

Operatori + mund te perdoret per te kombinuar dy vargje (string).

let dita = ‘e diele’;  
console.log(‘Sot eshte dite ‘ + dita + ‘.’); // rezultati: Sot eshte dite e diele.

Ne shembullin e mesiperm, ne caktuam vleren ‘e diele’ te variabli dita. Ne rreshtin e dyte, operatori + eshte perdorur per te kombinuar tre vargjet: ‘Sot eshte dite ’, vleren e ruajtur ne dita dhe ‘.’. Shembulli printon ne tastiere : Sot eshte dite e diele.

Ne versionin ES6 te JavaScript, ne mund te fusim variabla brenda vargjeve duke perdorur template literals. Ky proces quhet String Interpolation. Shikoni shembullin me poshte:

const dita = ‘e diele’;  
console.log(`Sot eshte dite ${dita}.`); // rezultati: Sot eshte dite e diele

Nje template literal definohet nepermjet simbolit `. Nje nga perfitimet me te medha qe sjell perdorimi i template literals eshte lexueshmeria e kodit.

**1.15 Operatori typeo**

Kur jeni duke koduar, mund te kete momente kur eshte e dobishme te kontrolloni llojin e te dhenes te nje variabli ne program. Nese ju duhet te kontrolloni llojin e te dhenave te vleres se nje variabli, mund te perdorni operatorin typeof.

Operatori typeof kontrollon vleren ne te djathte te tij dhe kthen nje varg te llojit te te dhenave.

const variable = ‘tekst’;  
console.log(typeof variable); // rezultati: string

const variable2 = 10;  
console.log(typeof variable2); // rezultati: number

Por nese periqeni te kontrolloni llojin e tipit null atehere rezultati do te jete “object”.

console.log(typeof null); // rezultati: object

Ky eshte nje gabim ne JavaScript, pasi null eshte nje tip primitive dhe jo objekt. Ky gabim nuk eshte korrigjuar, pasi nese do te korrigjohej shume prej aplikacioneve ekzistente nuk do te funksiononin.