# Változók

## Bevezetés

Ebben a leckében egy téglalap kerületét és területét számító programot készítünk. Közben megtanuljuk a változókat, a beolvasást és a kerekítést.

## Változók

A változók a programokban értékek tárolására szolgálnak. Ezek az értékek lehetnek számok, szövegek, vagy más típusú adatok is.

A változókat használat előtt deklarálni kell. Ennek formája a JavaScriptben:  
let nev;

A változó neve betűvel, aláhúzással vagy dollárjellel kezdődhet, utána állhatnak további betűk, aláhúzások, dollárjel vagy számjegyek. Például: a, a\_oldal, aOldal, oldal1, …  
A kis- és a nagybetűk különbözőnek számítanak! Szóköz vagy ékezetes betű nem lehet a nevekben! A név nem lehet foglalt szó (let, function, …). Sőt, a változók neve ne egyezzen meg a HTML fájlban használt azonosítókkal!

A változó típusát (szám, szöveg, …) nem kell megadni, azt majd a hozzá rendelt érték dönti el. (Amíg nem rendelünk hozzá értéket, addig undefinied lesz az értéke.)

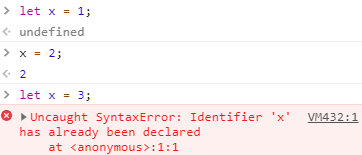
A változóhoz értékadó utasítással rendelhetünk értéket. Ennek formája:   
nev = kifejezés;

A változóba a kifejezés értéke kerül. Egy változóhoz többször egymás után is rendelhetünk értéket, mindig az utolsó értékadás eredményét tárolja.

A deklarációt és az első értékadást össze lehet vonni így:  
let nev = kifejezés;

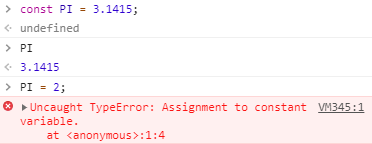
## Változó értékének módosítása

Egy változó értékét bármikor módosíthatjuk, de a változót nem deklarálhatjuk újra. Például:



Ha egy változónak csak egyszer adunk értéket a programban, és később már nem szeretnénk módosítani, akkor a let kulcsszó helyett a const kulcsszót célszerű használni:

Például:



Ha az állandó értékét már a program futása előtt tudjuk, akkor nagy betűvel szokás írni a nevét, ha csak a program futása közben kap értéket, akkor kisbetűvel.

## Adattípusok

Mint már mondtuk, egy változó típusát a hozzá rendelt érték dönti el. Ugyanahhoz a változóhoz egymás után többféle típusú értéket is rendelhetünk. (Például először számot, azután szöveget.)

A JavaScriptben a következő adattípusok vannak:

* Number: egész vagy tört szám (13, 6.23)  
  A matematikai műveletek a szokásos módon végezhetők velük.
* String: szöveg ("valami", ’valami’, ’A’)  
  Nincs külön karakter típus!
* Boolean: logikai érték (true, false)
* undefinied: már deklaráltuk, de még nem kapott értéket
* null: üres, nincs értéke – de ezt mi állítottuk be
* Object: összetett objektum (ld. később)

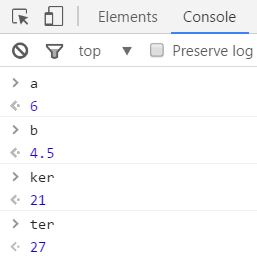
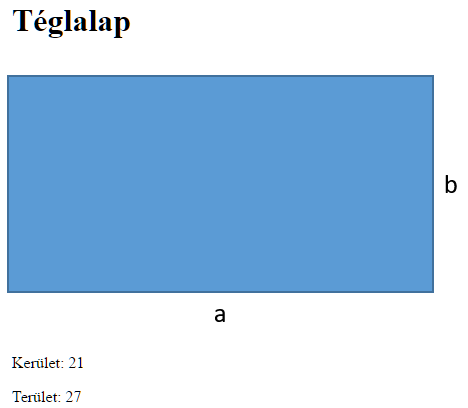
A műveletek és a függvények szükség szerint módosítják a típusokat. Például az alert() függvény a számokat stringekké alakítja.

## Előkészületek

1. Nyisd meg a *tegla* mappát, majd az abban lévő fájlokat a Visual Studio Code-ban!
2. Nyisd meg az oldalt a böngészőben is!
3. Nézd át az eddig elkészített kódokat!

## Példa

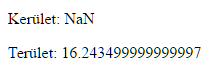
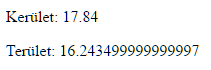
Példánkban a téglalap két oldalát szeretnénk egy-egy változóban tárolni. Tegyük fel hogy az a oldal hossza 6, a b oldalé pedig 4 és fél. A törteket tizedesponttal jelöljük (4.5). Az a és a b változók szám (Number) típusúak lesznek.

1. Írd a következőket a *szamit.js* fájlba:  
   
2. Ezután kiszámítjuk a kerületet és a területet. Folytasd így a scriptet:  
     
   Milyen típusúak a ker és a ter változók?
3. Mentsd a scriptet, frissítsd az oldalt a böngészőben, majd kapcsold be a fejlesztői eszközöket! Ezután írasd ki a konzolon a változók értékeit:  
   
4. A ker változó értékét a kerulet azonosítójú bekezdésbe szeretnénk kiíratni. Ezt így tehetjük meg:  
     
   Próbáld ki! Figyeld meg, hogy a JavaScript automatikusan átalakítja a számot szöveggé!
5. Szebb lenne az eredmény, ha a szám elé odaírnánk, hogy mi az. Tegyük meg:  
     
   Próbáld ki! A JavaScriptben a + jellel fűzhetünk össze szövegeket. A program az összefűzés előtt a ker értékét szöveggé alakítja.
6. Készítsd el önállóan a terület kiírását, majd próbáld ki! Minta:  
   

## Beolvasás

Az a és a b változók fix értékei nagyon korlátozzák a program használhatóságát, ezért inkább beolvastatjuk a programmal az értéküket.

A beolvasásra a prompt() függvényt használhatjuk. Ennek két paramétere van, a kérdés szövege és az alapértelmezett válasz.

1. Próbáld ki a következőket a konzolon:  
     
   Figyeld meg, hogy a beolvasott érték szöveg, de ha számítást végzünk vele, akkor automatikusan számmá alakul!
2. Írd át a script elejét így:  
   
3. Próbáld ki törtszámokkal! Mi lehet a probléma a kerületnél?  
   
4. Írasd ki a konzolon a következőket:  
     
   Látod már a problémát? A NaN (Not a Number) érték akkor jelenik meg, ha nem számot kap a program.
5. Hogyan lehetne megoldani a problémát? Egy egyszerű megoldás, ha a beolvasásnál a kifejezés elé teszel egy + jelet. Próbáld ki!  
   
6. Ennél elegánsabb megoldás, ha a Number() függvényt használod:   
     
   Próbáld ki ezt is:  
     
   Mi történik, ha nem számot adsz meg?

## Kerekítés

Nézzük meg, hogyan kerekíthetjük az eredményeket!

1. Ha egészekre szeretnél kerekíteni, akkor használd a Math.round() függvényt:  
     
   Próbáld ki!
2. Ha mindig két tizedesjeggyel szeretnéd megjeleníteni az eredményt, akkor használd a .toFixed(2) metódust:  
     
   Próbáld ki!

## Kiegészítések

A JavaScript nyelvet is állandóan fejlesztik. Az ECMAScript 2015 vagy más néven JS6 szabványban vezették be a var és a const kulcsszavakat.

A korábbi programokban a var kulcsszót használták a let helyett, állandó megadására pedig nem volt lehetőség.

A let a var-tól a hatókörben és néhány más dologban is eltér (ld. később).

Vannak olyan programozók,, akik még a var szót használják, és vannak olyanok, akik már áttértek a let és a const használatára (és ez lesz a jövő).