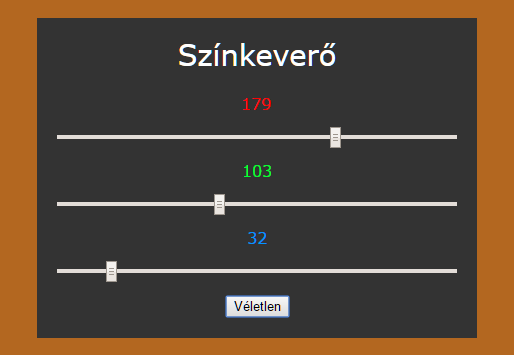
# Színkeverő

## Bevezetés

Ebben a leckében egy olyan programot készítünk, amelyben az alapszínek értékeinek módosításával tetszőleges színeket kikeverhetünk.



A csúszkákkal beállított szín az oldal hátterében jelenik meg. A Véletlen gomb mindhárom összetevőt véletlen értékre állítja.

Gondold végig, milyen HTML elemek vannak az oldalon!

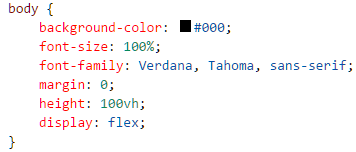
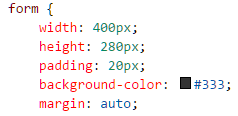
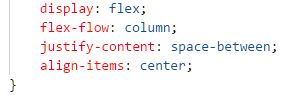
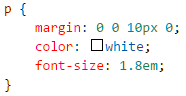
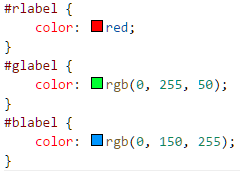
## Előkészületek

1. Nyisd meg a *szinkevero* mappát, majd az abban lévő fájlokat a Visual Studio Code-ban!
2. Nyisd meg az oldalt a böngészőben is!
3. Nézd át és értelmezd az eddig elkészített HTML kódot!
4. Figyeld meg, hogy most a label elem másik formáját használtuk, mint korábban, és a csúszkákat az input elem range típusával készítettük el! Milyen határok között változhat a csúszkák értéke?

## Formázás

Gondold végig, milyen formázási beállításokra van szükség az egyes elemeknél!

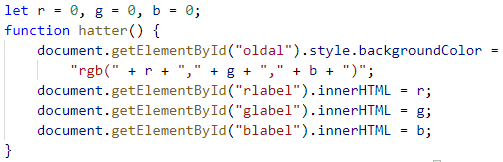
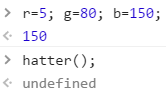
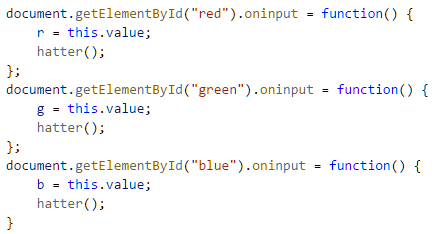
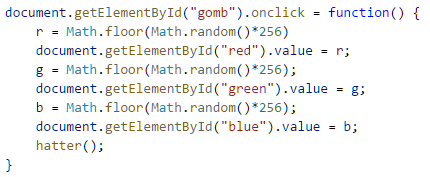
A következő formázásokat a *szinek.css* fájlban végezd el, majd próbáld ki őket különböző ablakszélességekkel!

1. Az oldal háttere induláskor fekete lesz. A betűméret az alapbeállításnak megfelelő (16px), a betűtípus pedig Verdana, Tahoma, vagy talpatlan. Margó nem kell, és az oldal magassága mindig akkora legyen, mint a viewporté! (Így az oldal nem lehet rövidebb, mint a rendelkezésre álló hely.) A szürke formot pontosan az oldal közepére szeretnénk elhelyezni, ezért az oldalt flex tárolóvá alakítjuk.  
   
2. A form 400 x 280 képpontos legyen, 20 képpontos belső margóval, háttérszíne pedig sötétszürke (#333)! Mivel pontosan az oldal közepére szeretnénk elhelyezni, margói legyenek automatikusak!  
   
3. A formon belüli elemeket egymás alá szeretnénk elhelyezni, egyenletes távolságra egymástól, ezért a formot is flex tárolóvá kell alakítani. Az elemeket vízszintesen középre kell igazítani:  
   
4. A form tetején lévő bekezdést így állítsd be:  
   
5. A csúszkákat vízszintes irányban szét kell húzni a formon belül:   
   
6. Végül az egyes címkék színeit be kell állítani az alapszíneknek megfelelően. Itt egy kicsit csalunk: a zöld és a kék nem az alapszín lesz, hogy jobban látható legyen.  
   

## A script elkészítése

A scriptben nyilván kell tartani, hogy éppen mennyi a három színösszetevő értéke. Erre szolgálnak az r, a g és a b globális változók.

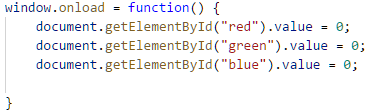
Ha valamelyik színösszetevő értéke megváltozik, akkor át kell állítani a háttérszínt, és be kell írni a címkékbe az új értékeket. Ezt a hatter() függvénnyel oldjuk meg.

1. Írd be a következőket a *szinek.js* fájlba:  
     
   Figyeld meg, hogyan állítjuk össze az rgb() függvényt!
2. Frissítsd az oldalt, majd próbáld ki a következőt a konzolon:  
     
   Figyeld meg, hogy a háttér színe és a címkék megváltoztak az új értékeknek megfelelően, de a csúszkák értéke nem változott!
3. írd be a következőt a konzolra, és figyeld meg a kék csúszka változását!  
     
   Hogyan lehetne a többi csúszka értékét is megváltoztatni?
4. A csúszkák értékének módosítása az egérrel vagy a billentyűzettel az oninput eseményt generálja a csúszkákon. Készítsd el, majd próbáld ki az oninput eseményeket kezelő kódot!  
     
   A this szóval az éppen kiválasztott elemet adhatjuk meg. Így másodszorra már nincs szükség a getElementById() metódusra. A csúszkáknál is a value adja meg a beállított értéket.
5. A véletlen gomb megnyomásakor az r, a g és a b értékét véletlenül választjuk meg 0 és 255 között. Ezután beállítjuk a csúszkák értékét ennek megfelelően. Végül meghívjuk a hatter() függvényt.  
     
   Próbáld ki!

## Önálló feladat

Sajnos a csúszkák kezelése a böngészőkben még nem tökéletes. Az Edge-ben és a Firefox-ban a csúszkák nem állnak be alaphelyzetbe frissítés után. Próbáld ki!

Ezt úgy oldhatod meg, hogy a szinek.js végére beírod az alábbiakat:



Az window.onload esemény akkor következik be, amikor a teljes oldal betöltődött. Próbáld ki!