# Stílusok kezelése JavaScriptben

Egy elem megjelenését három szinten tudjuk befolyásolni stílusbeállításokkal:

1. az elem style attribútumán keresztül,
2. az elem class attribútumán keresztül, valamint
3. az oldalba illesztett stíluslapokkal.

JavaScriptben a **style** attribútumnak az elemnek megfelelő DOM objektum **style** tulajdonsága felel meg.

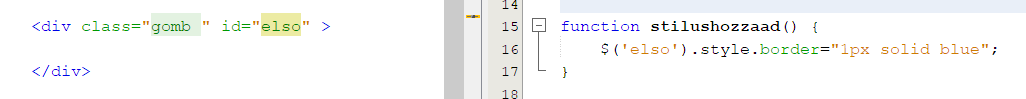
Íme néhány példa az átírásra:

|  |  |
| --- | --- |
| **CSS tulajdonság** | **JavaScript tulajdonság** |
| color | color |
| background | background |
| margin-left | marginLeft |
| font-size | fontSize |
| border-bottom-width | borderBottomWidth |

Ennek megfelelően egy tetszőlegesen kiválasztott elem stílusbeállításait style objektuma tulajdonságainak megfelelő értéket adva lehet megtenni. Például egy elem alsó keretének vastagságát a következőképpen lehet beállítani:

###### elem.style.borderBottomWidth = '5px';

Péda:



## Számított stílus

**A böngésző minden elemről pontosan tudja, hogy hol helyezkedik el, és hogyan néz ki, azaz ismeri az összes elem összes stílustulajdonságát, azt is, amit nem adtunk meg sem a css-ben, sem a js-ben. Ezt az információt hívjuk számított stílusnak.**

Ennek elérése szabványos metódussal:

###### window.getComputedStyle(elem)

|  |  |
| --- | --- |
|  | **A style tulajdonságon keresztül:** |
| **Számított stílus alapján:** |

## A class attribútum programozása

A HTML elemek class attribútumát a nekik megfelelő DOM objektum className tulajdonsága alatt lehet lekérdezni és módosítani.

html:

###### <elem id="elem2" class="osztaly1 osztaly2 osztaly3"></elem>

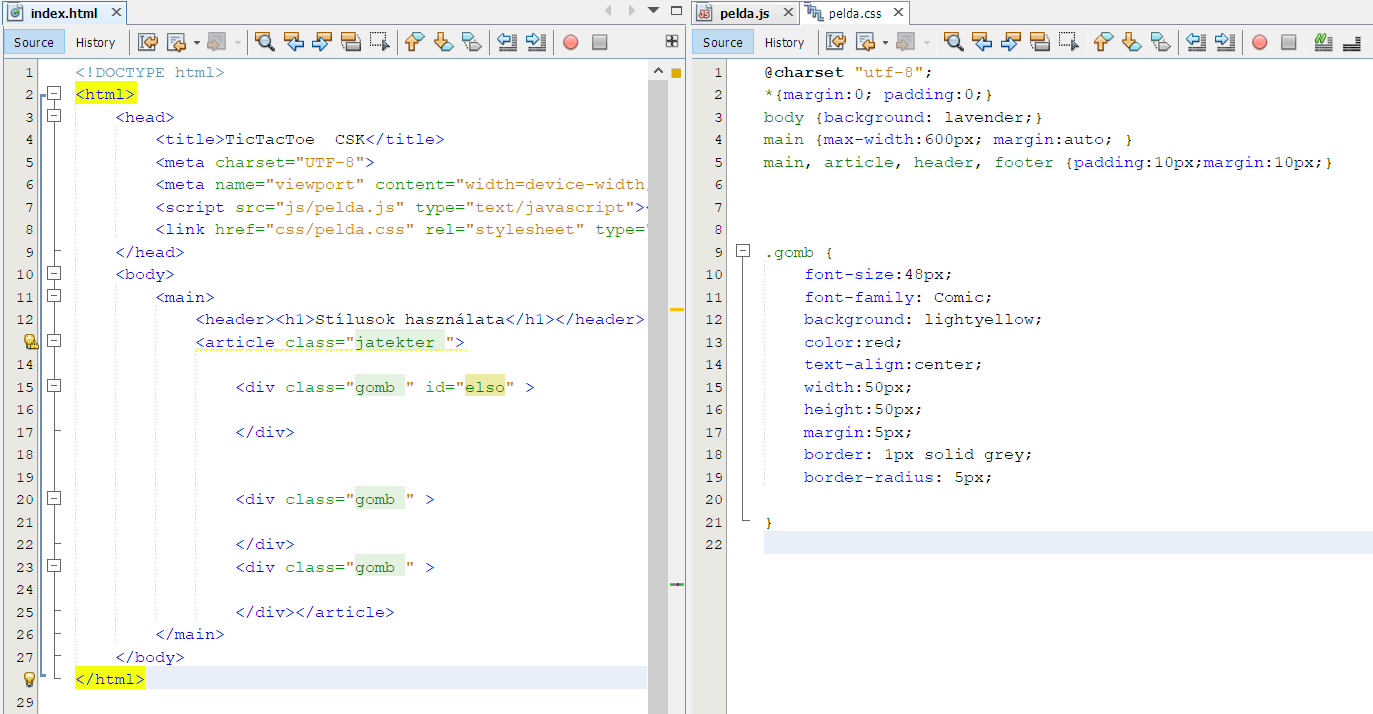
JS:

###### document.getElementById('elem2').className;//osztaly1 osztaly2 osztaly3

## Elemek elérése class tulajdonságuk alapján

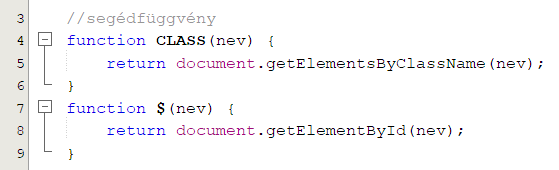
**Az id-vel való eléréshez hasonlóan történik. A különbség abban áll, hogy míg id azonosítóból csak egyetlen lehet egy HTML oldalon belül, addig a class névvel elem több is lehet. Ezeket egy tömbbe gyűjti a JS, és indexel hivatkozva lehet elérni. Az elemeket a for ciklussal érdemes bejárni.**

Az alábbi példakód 3 div elemhez ugyanahhoz a gomb osztályhoz vannak rendelve.

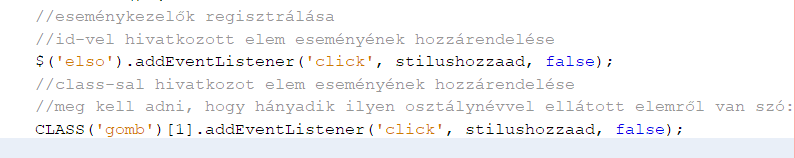


Nézzük, hogy hivatkozzuk rájuk a JS kódban!

Érdemes ezekhez is egy rövidítő segédfüggvényt bevezetni, az id-hez hasonlóan:



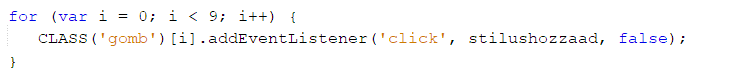
A kattintás esemény hatására hívjuk meg ***stilushozzaad()*** függvényt:



#### A 0 indexű elem lesz az első, az 1. indexű a második a sorban.

Az első elemre hivatkozhatok a firstChilde értékkel is.

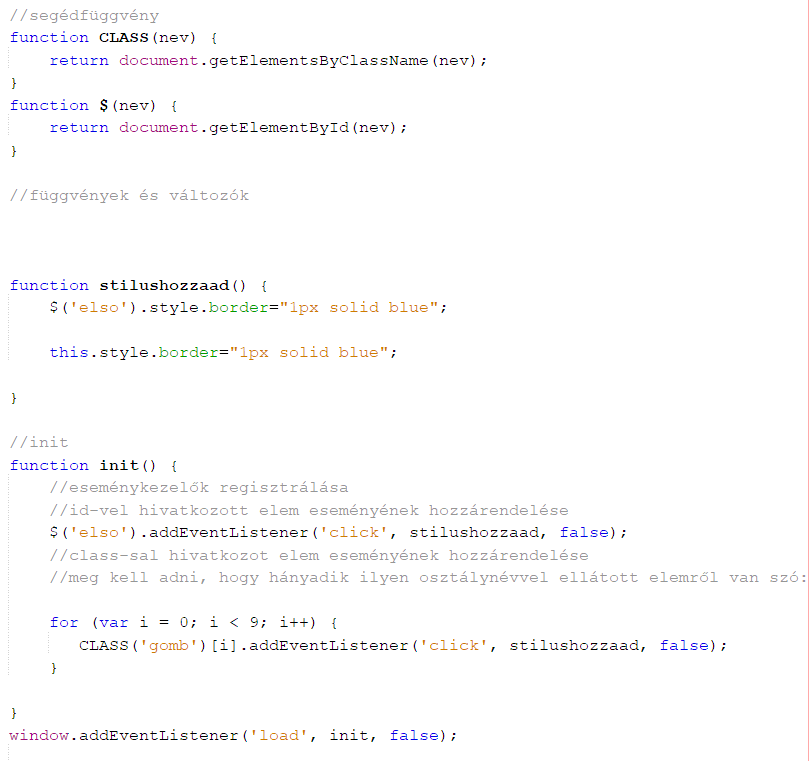
Ha az összes ilyen osztálynévvel ellátott elemhez hozzá akarom rendelni az eseményt, akkor ciklusban hívom meg az eseménykezelő regisztrálását.



Ezután az eseménykezelő által meghívott függvényben az aktuális elemre a this -szóval tudok hivatkozni.



A JS kódunk most így néz ki:



Ha több stílustulajdonságot is meg kell adnunk a JS kódban, az nagyon nehézkes lehet. Sokkal egyszerűbb, ha a css-ben elkészítünk egy class definíciót, és ezt rendeljük hozzá az adott elemhez.

|  |  |
| --- | --- |
| Íme a css tartalma: | A js-ben ezt az osztályt az alábbimódon rendelhetjük a gombjainkhoz: |

A kódunk hatására, amennyiben a div elemek által implementált gombok bármelyikére kattintok, megváltozik azoknak a háttér és betűszíne.

#### Osztály hozzáadása, törlése vagy meglétének ellenőrzése egy elemnél – a profi megoldás

E feladatok gyakorlatilag a className által visszaadott szöveg feldolgozásából állnak. Hozzáadáskor egy szóközzel elválasztva kell hozzáfűzni az eddig még nem létező osztálynevet, törléskor meg kell keresni, és törölni kell, meglétének ellenőrzésekor pedig a keresés eredményét kell visszaadni. A keresést legegyszerűbb reguláris kifejezéssel megadni.

Példa:

###### function vaneOsztaly(elem, o) {

###### return elem.className.search(new RegExp('(^|\\s+)' + o + '(\\s+|$)')) > -1;

###### }

###### function hozzaadOsztaly(elem, o) {

###### if (!vaneOsztaly(elem, o)) {

###### elem.className += (elem.className ? ' ' : '') + o;

###### }

###### }

###### function torolOsztaly(elem, o) {

###### elem.className = elem.className

###### .replace(new RegExp('(^|\\s+)' + o + '(\\s+|$)'), ' ')  //osztálynév törlése

###### .replace(/^\s\*|\s\*$/g, '');     //kétoldali felesleges szóköz eltávolítása

###### }

Forrás: <http://webprogramozas.inf.elte.hu/tananyag/wf2/lecke10_lap1.html#fejezetek>

Stíluslapok használatáról és az animációkról a fenti forrásban többet is olvashatsz.

# Feladat leírása: Tic-Tac-Toe játék elkészítése

Készítsük el az alábbi játékot:

Figyeljünk a mappaszerkezetre: Külön mappába kerüljön a css és külön mappába a js fájlok!



header

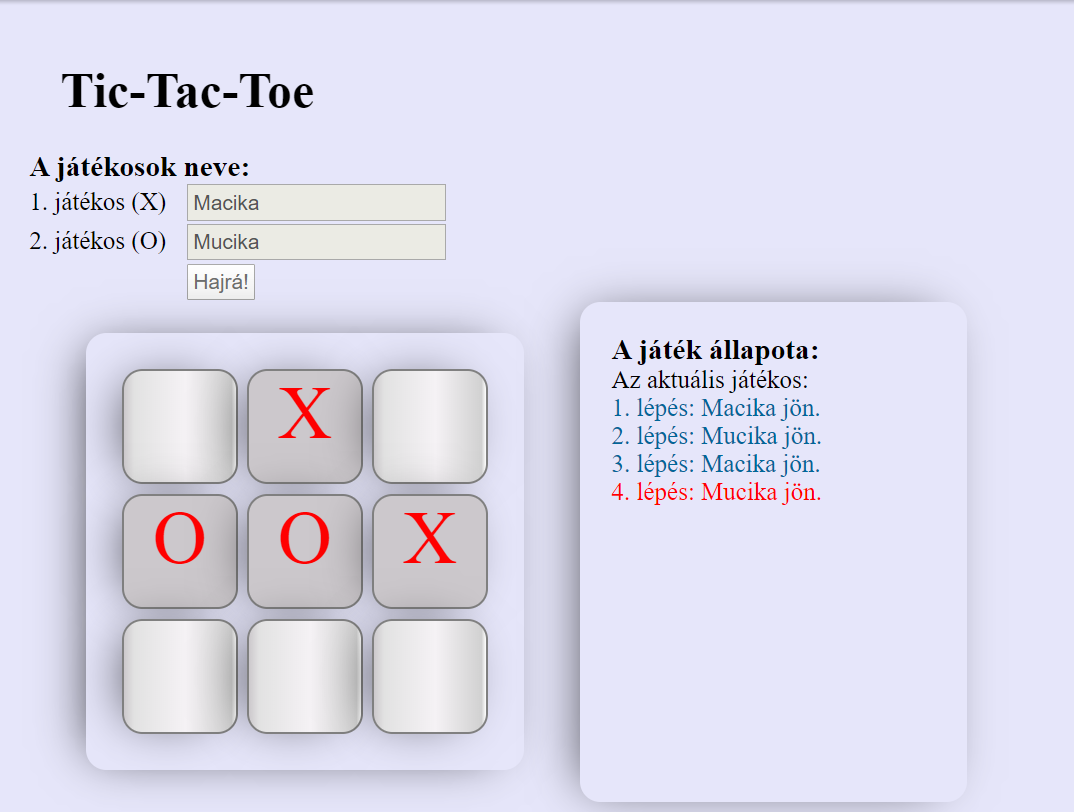
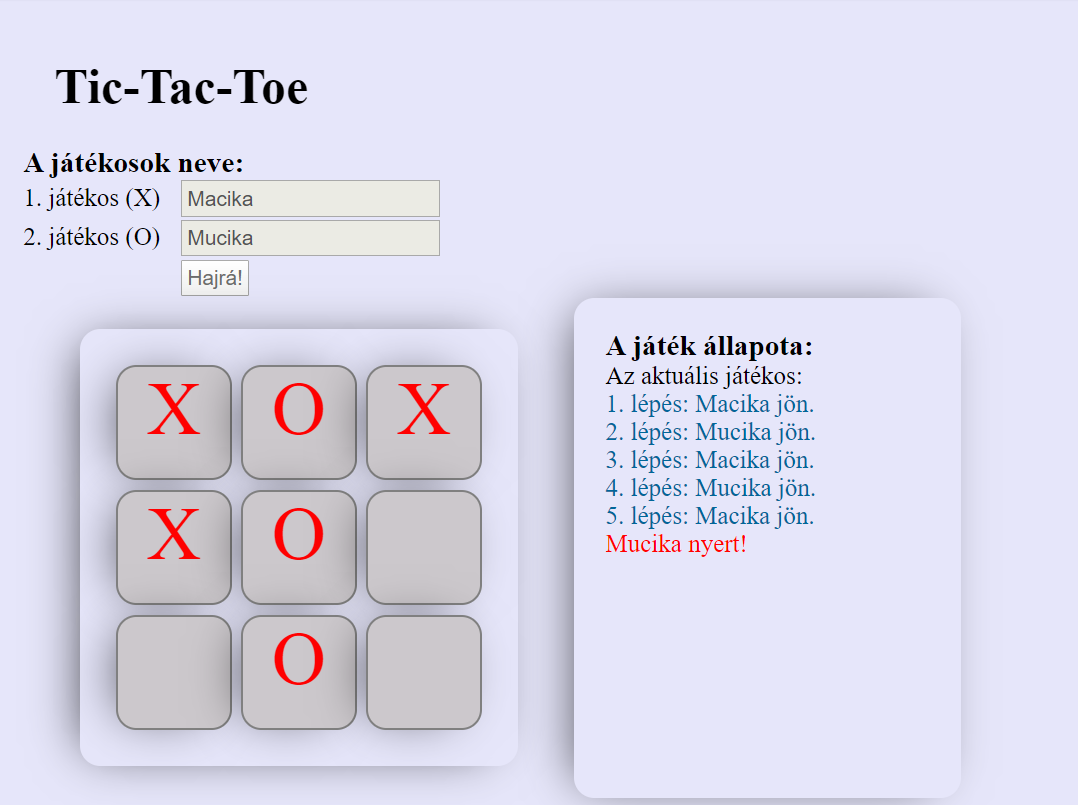
aside

article

section

**A HTML szerkezete:**

1. A fejlécben legyen a játék neve.
2. Ezután következik egy section elem, ez tartalmazza az űrlapelemeket.
3. Ezután a játéktér: a 9 div-vel.
4. Majd az aside elem, ide írjuk majd ki a játék állapotát.
5. A footerbe kerüljön a készítő neve

##### section:

1. Két szöveges mezőbe lehessen megadni a játékosok nevét. Legyen alapértelmezett név is.
2. Továbbá legyen egy gomb (button), lásd el id-vel (kezd), és rendelj hozzá kattintás eseményt. Kattintásra hívja meg a játékkezd() függvényt.

##### article:

1. Ez lesz a játéktér: 9 div, a „***gomb***” osztálynévvel ellátva. css-ben formázva, griddel a képen láthatóan elrendezve.
2. Rendeljünk hozzájuk kattintás eseményt.

* Ha kattintunk, akkor a div tartalma változzon meg: Kerüljön bele X, vagy O betű.
* Továbbá változzon meg a háttérszín, és többé már ne lehessen rákattintani! (cursor:not-allowed;)

1. Rendeljünk a gombokhoz (mouseover) eseményt!

* Ha a gombok fölé viszem az egeret, változzon meg a szegély színe! Ezt most JS segítségével oldjuk meg. az egérmozgatásra rendeljünk hozzá függvényt, melyben megváltoztatjuk a gomb valamelyik formázási tulajdonságát!

1. Rendeljünk a gombokhoz (mouseout) eseményt!

* Ha az egérrel távozunk a gombról (mouseout), akkor álljon vissza az eredeti formátum!

A **játékkezd()** függvény:

1. Tegye inaktívvá a input mezőket és a Kezdhetjük? feliratú gombot is, a gomb új felirata Hajrá! legyen.
2. Kezdetben ne látszódjon a 9 gomb és a jobb oldali eredmény mező sem. A játékkezd() függvény tegye ezeket láthatóvá!

**+További feladatok:**

A játékosok felváltva lépnek. A jobb oldali (aside) elemben írjuk ki, hogy éppen ki a soron következő játékos, és a div elemekbe ennek megfelelően kerüljön a X, vagy a O.

A szövegben mindig az utolsó elem (lastChild) elem legyen piros, a többi szöveg kék színű legyen. (stíluslapok hozzáadásával oldjuk meg)

**++ A végén írja ki, hogy ki a nyertes, illetve, ha döntetlen játék!**

##### Figyeljünk arra, hogy a js fájl szerkezete a tanultaknak megfelelő legyen! Ilyesmi…

