

Vászon a weblapon

A versenyen a mintakódok és a feladatok többnyire rajzoláshoz fognak kapcsolódni. Hogyan lehet egy weblapra rajzolni? A megoldást a HTML5 szabványban bevezetett [canvas](#) (vászon) jelenti. Erre tudtuk majd rajzolni, előtte viszont meg kell adni a HTML kódban a méretét, azonosítóját, egyéb tulajdonságait. Ez nem bonyolult, mivel ez is "csak" egy HTML utasítás.

```
<canvas id="vaszon" width="200" height="200"></canvas>
```

Az így kapott vászon természetesen nem látszik a weblapon, mivel nem adtunk meg neki keretet. Ha mégis szeretnék ilyet, akkor meg kell adnunk a vászon stílusát a következő módon:

```
<canvas id="vaszon" width="200" height="200" style="border:1px solid #000000;"></canvas>
```

Így már látszik az 1 pixel vastag fekete keret. Rajzolni rá már kicsit bonyolultabb. Nézzétek át a következő oldal kódját!

```
<!DOCTYPE html>
<html>
<body>
<canvas id="vaszon" width="200" height="200" style="border:1px solid
#c3c3c3;"></canvas>
<script>
    var c = document.getElementById("vaszon");
    var ctx = c.getContext("2d");
    ctx.moveTo(0,0);
    ctx.lineTo(200,200);
    ctx.stroke();
</script>
</body>
</html>
```

A vászon 200x200-as, 1 pixel vastag "szürkés" kerettel. A vászonra rajzoltunk egy szakaszt. Először megadtuk a kezdőpontját, ([moveTo](#)(0,0)), majd ezt a pontot összeköttöttük a (200,200) ponttal a [lineTo](#) segítségével. A szakasz csak a [stroke](#)() hatására fog megjelenni. A rajzolás kódját a `<script></script>` tag-ek közé kellett rakni. A közötté lévő utasítások adják a rajzolás kódját JavaScript nyelven. De mi az a két sor, ami **var**-ral kezdődik? A továbblépéshez megmutatjuk, hogyan tudtok JavaScript kódot írni a HTML utasítások közé!

JavaScript

Ha a HTML kódban számításokat akarunk végezni, vagy néhány feltétel teljesülését szeretnénk megvizsgálni, esetleg utasításokat többször végrehajtani, akkor adat- és vezérlőszervezetekre lesz szükségünk. A HTML-ben ilyen utasítások nem léteznek. Egy megoldás lehet erre a [JavaScript](#) kódok beágyazása. Ilyenkor a `<script>` és a `</script>` tag-ek közé kell írunk a JavaScript utasításokat, melyeket a böngésző program fog értelmezni és végrehajtani. Visszatérve az előző példában látott kódra: a **változót** névvel azonosított **memóriaterületnek** tekintjük, különböző értékek tárolásának és feldolgozásának az eszköze. A változókat a [var](#) kulcsszó után deklaráljuk. Értékadó kifejezésekben az egyenlőségjel bal

oldalán álló változó veszi fel a jobb oldalon álló kifejezés értékét. A kifejezések kiértékelését a JavaScript interpreter (értelmező) végzi el. Az eredmény a kifejezésben használt változók és a kijelölt műveletek jellegétől függően leggyakrabban [szám, szöveg vagy logikai](#) típusú. Az előző kódban két változónk volt: egy **c** nevű és egy **ctx** nevű. Az elsőben a weblap egyik objektumát, a vásznat "kértük" le. Innentől kezdve a c objektummal tudjuk a vásznat elérni, hivatkozni rá. ("Vászon! Mától kezdve a neved c!"). A ctx pedig egy olyan objektumot fog tárolni, melyek a vásznon történő rajzoláshoz ad [metódusokat](#) (eljárásokat és függvényeket) és tulajdonságokat. Innentől kezdve tehát a ctx. után írhatunk olyan metódusokat (eljárásokat és függvényeket), melyekkel rajzolni tudunk a vászon nevű canvas-ra, amelyet a **c változón** keresztül tudunk elérni. A későbbiekben további JavaScript utasításokkal fogtok megismerkedni. Most azonban térjünk vissza a rajzoláshoz!

Rajzolás, rajzolás, rajzolás

```
<!DOCTYPE html>
<html>
<body>
<canvas id="vaszon" width="400" height="400" style="border:1px solid
#c3c3c3;"></canvas>
<!-- a vászon azonosítója, szélessége, magassága, 1 px keret a megadott
kóddal -->
<script>
    var c = document.getElementById("vaszon");
    var ctx = c.getContext("2d");
    ctx.moveTo(0,0);
    ctx.lineTo(200,200);
    ctx.stroke(); // szakasz rajzolás

    ctx.rect(20,20,150,100);
    ctx.stroke(); //téglalap rajzolása

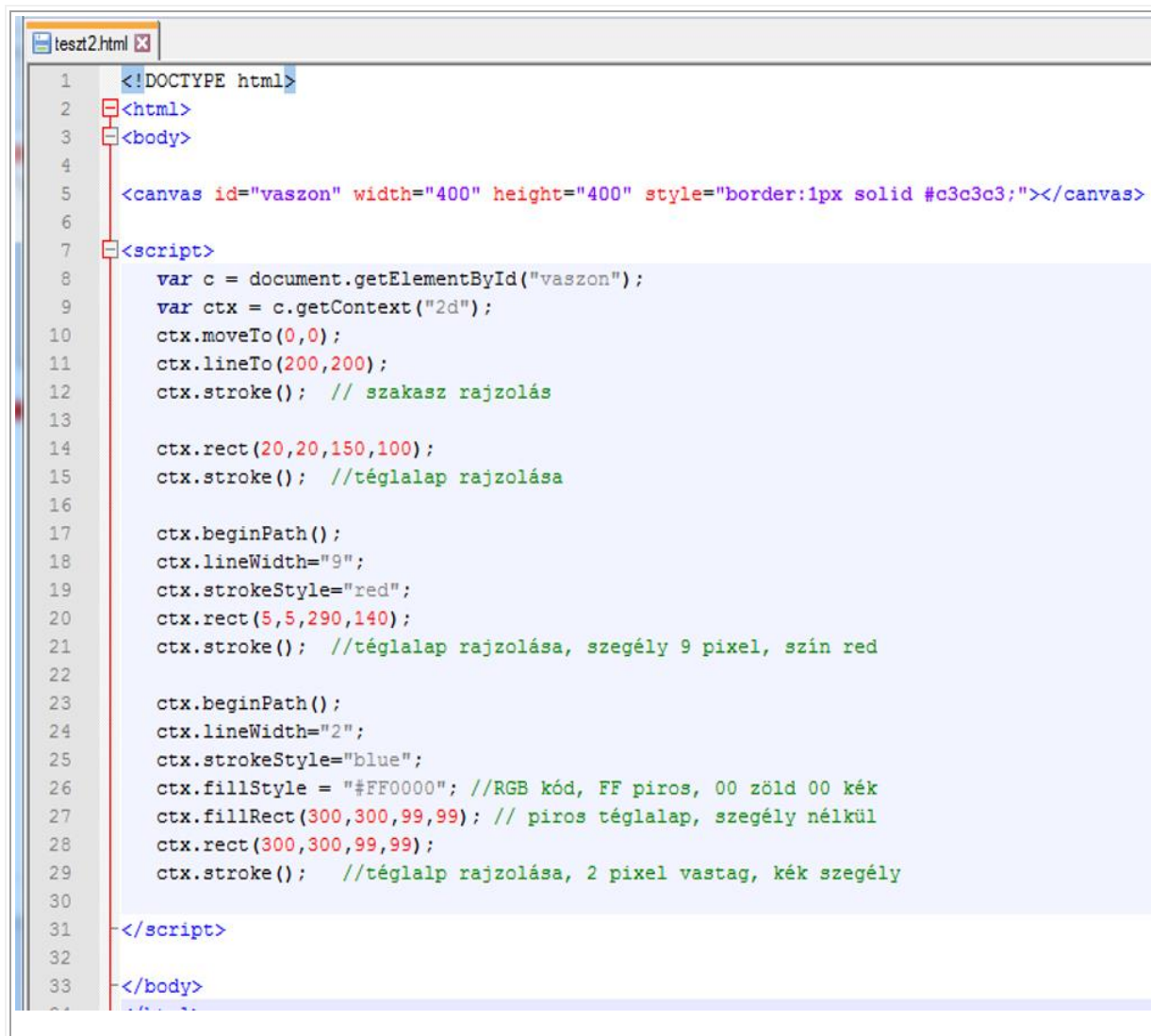
    ctx.beginPath();
    ctx.lineWidth="9";
    ctx.strokeStyle="red";
    ctx.rect(5,5,290,140);
    ctx.stroke(); //téglalap rajzolása, szegély: 9 pixel, szín:red

    ctx.beginPath();
    ctx.lineWidth="2";
    ctx.strokeStyle="blue";
    ctx.fillStyle = "#FF0000"; //RGB kód, FF piros, 00 zöld 00 kék
    ctx.fillRect(300,300,99,99); // piros téglalap, szegély nélkül
    ctx.rect(300,300,99,99);
    ctx.stroke(); //téglalap rajzolása, 2 pixel vastag, kék szegéllyel

</script>
</body>
</html>
```

A fenti kódban rajzoltunk egy szakaszt az alapértelmezett tulajdonságokkal (keret vastagság, szín), majd egy téglalapot, melynek bal felső sarka (20, 20), szélessége 150 pixel, magassága 100 pixel. Az origó a vászon bal felső sarka, ami egyben a képernyő bal felső sarka is. Szintén nem változtattuk meg az alapértelmezett tulajdonságokat. A következő téglalap már 9 pixel [vastag, piros](#) színű, 290x140-es méretű. Majd két téglalapot rajzoltunk. Az első pirossal van [kitöltve](#), és a vászon jobb alsó sarkában (300, 300) van, 99x99-es. A másik pedig 2 vastag, kék színű és pont akkora, mint a kitöltött. Így olyan, mintha kék szegélyű piros téglalapunk lenne.-) A // pedig a megjegyzés jele. Az utána lévő szöveget a böngészők nem értelmezik, mintha ott sem lenne.

Ha kimásoljátok a kódot a NotePad++-ba és elmentitek, kb. ezt kellene látnotok:



```
1 <!DOCTYPE html>
2 <html>
3 <body>
4
5 <canvas id="vaszon" width="400" height="400" style="border:1px solid #c3c3c3;"></canvas>
6
7 <script>
8     var c = document.getElementById("vaszon");
9     var ctx = c.getContext("2d");
10    ctx.moveTo(0,0);
11    ctx.lineTo(200,200);
12    ctx.stroke(); // szakasz rajzolás
13
14    ctx.rect(20,20,150,100);
15    ctx.stroke(); //téglalap rajzolása
16
17    ctx.beginPath();
18    ctx.lineWidth="9";
19    ctx.strokeStyle="red";
20    ctx.rect(5,5,290,140);
21    ctx.stroke(); //téglalap rajzolása, szegély 9 pixel, szín red
22
23    ctx.beginPath();
24    ctx.lineWidth="2";
25    ctx.strokeStyle="blue";
26    ctx.fillStyle = "#FF0000"; //RGB kód, FF piros, 00 zöld 00 kék
27    ctx.fillRect(300,300,99,99); // piros téglalap, szegély nélkül
28    ctx.rect(300,300,99,99);
29    ctx.stroke(); //téglalp rajzolása, 2 pixel vastag, kék szegély
30
31 </script>
32
33 </body>
```

Már csak egyetlen olyan metódus (függvény vagy eljárás) van, amelyről nem szóltunk egy szót sem. Ez pedig a [beginPath\(\)](#). Erre azért van szükség, mert több alakzatot rajzolunk eltérő paraméterekkel. Ilyenkor ezzel az metódussal jelezzük, hogy új "útvonalat" kezdünk, vagy esetleg az aktuálisat akarjuk újakezdeni.

Eddig egyenes vonalakkal építettük fel "ábránkat". Van lehetőségünk görbe vonalakra is. Egyik ilyen a körív rajzolás, amellyel teljes kört is tudunk rajzolni.

```
<!DOCTYPE html>
<html>
<body>
<canvas id="vaszon" width="400" height="400" style="border:1px solid
#c3c3c3;"></canvas>
<script>
    var c = document.getElementById("vaszon");
    var ctx = c.getContext("2d");
    ctx.beginPath();
    ctx.arc(95,50,40,0,2*Math.PI);
    ctx.stroke();
```

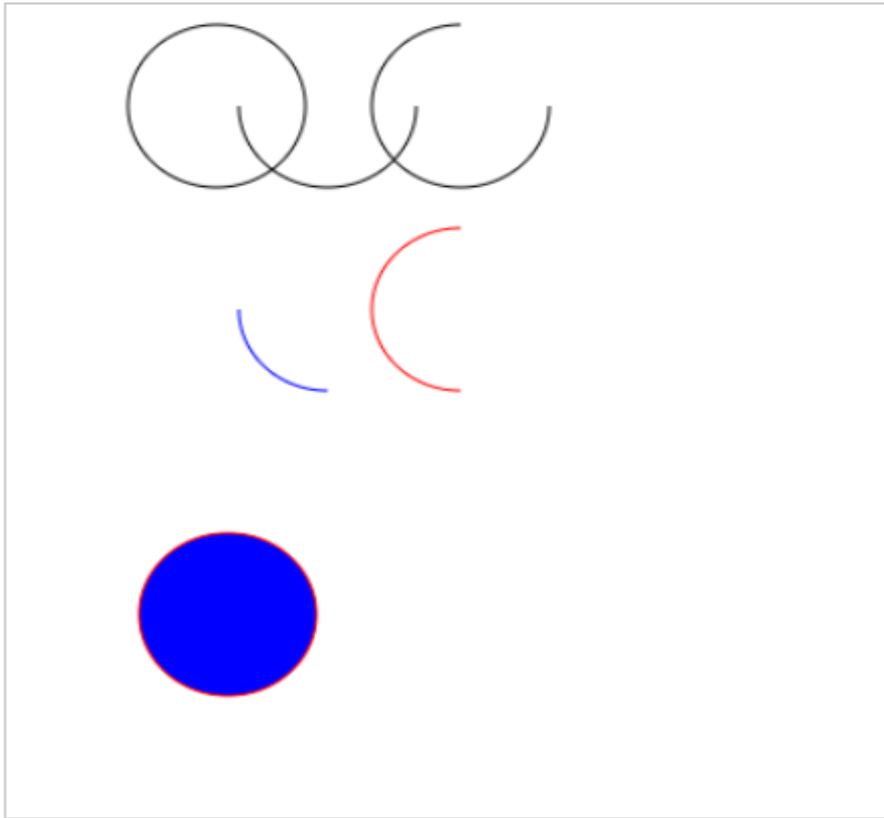
```
ctx.beginPath();
ctx.arc(145,50,40,0,1*Math.PI);
ctx.stroke();

ctx.beginPath();
ctx.arc(205,50,40,0,1.5*Math.PI);
ctx.stroke();

ctx.beginPath();
ctx.strokeStyle="#0000FF";
ctx.arc(145,150,40,0.5*Math.PI,1*Math.PI);
ctx.stroke();
ctx.beginPath();
ctx.strokeStyle="#FF0000";
ctx.arc(205,150,40,0.5*Math.PI,1.5*Math.PI);
ctx.stroke();
ctx.beginPath();
ctx.arc(100,300,40,0,2*Math.PI);
ctx.fillStyle = "blue";
ctx.fill();
ctx.stroke();

</script>
</body>
</html>
```

Az első körnek a középpontja a (95,40) pontban van, sugara 40. Az [arc](#) metódus (függvény vagy eljárás) 4. paramétere 0, azaz a kezdő szög radiánban 0, a végső szög pedig 360 fok, csak radiánban adtuk meg, azaz 2π . (Így ki is jön a teljes kör.) A következő körnek csak a felét rajzoltuk meg, a 3. körnek a 3/4-et rajzoltuk meg, mivel a végső szög 1.5π , a 4. körnek csak a negyedét rajzoltuk meg úgy, hogy a kezdőszög nem 0, hanem 0.5π , ráadásul kék színnel. Az 5. körnek ismét a felét rajzoltuk meg, de most a kezdőszög 0.5π , a vége pedig 1.5π és persze pirossal. A 6. kör pedig kék színnel van kitöltve, a körvonal szegélye pedig maradt a korábban (5. kör) beállított piros. Ezt kellene látnotok, ha kimásoljátok a kódunkat:



A következő kódban az előző köröket "felcímkezzük"! Megmutatjuk, hogy tudtok [szöveget rajzolni](#) a vászonra!

```
<!DOCTYPE html>
<html>
<body>
<canvas id="vaszon" width="400" height="400" style="border:1px solid
#c3c3c3;"></canvas>
<script>
    var c = document.getElementById("vaszon");
    var ctx = c.getContext("2d");
    ctx.beginPath();
    ctx.arc(95,50,40,0,2*Math.PI);
    ctx.font = "12px Arial";
    ctx.fillText("1. kör",95,50);
    ctx.stroke();
    ctx.beginPath();
    ctx.arc(145,50,40,0,1*Math.PI);
    ctx.font = "12px Times";
    ctx.fillText("2. ív",145,50);
    ctx.stroke();
    ctx.beginPath();
    ctx.arc(205,50,40,0,1.5*Math.PI);
    ctx.font = "13px Times";
    ctx.fillText("3. ív",205,50);
    ctx.stroke();
    ctx.beginPath();
    ctx.strokeStyle="#0000FF";
    ctx.arc(145,150,40,0.5*Math.PI,1*Math.PI);
    ctx.font = "16px Times";
    ctx.strokeText("4. ív",115,150);
    ctx.stroke();
</script>
</body>
</html>
```

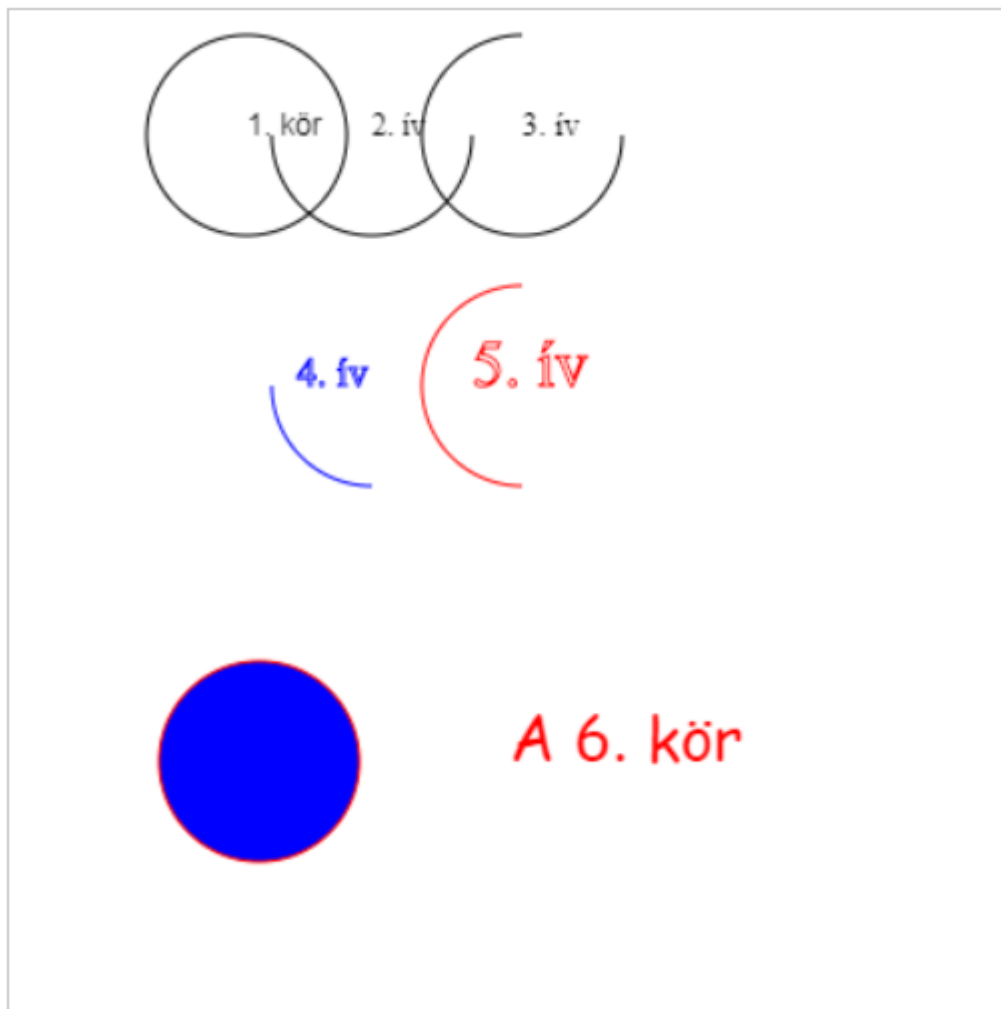
```

ctx.beginPath();
ctx.strokeStyle="#FF0000";
ctx.arc(205,150,40,0.5*Math.PI,1.5*Math.PI);
ctx.font = "26px Times";
ctx.strokeText("5. ív",185,150);
ctx.stroke();
ctx.beginPath();
ctx.arc(100,300,40,0,2*Math.PI);
ctx.fillStyle = "blue";
ctx.fill();
ctx.font = "25px Comic Sans MS";
ctx.fillStyle = "red";
ctx.fillText("A 6. kör",c.width/2,300);
ctx.stroke();

</script>
</body>
</html>

```

A megoldást tehát a [fillText](#) és a [strokeText](#) metódusok adják. Az előző kitölti a megadott színnel a karaktereket, az utóbbi csak a körvonalait rajzolja meg, ahogy a következő ábrán is láthatjátok. A **c.width** pedig megadja a válaszon szélességét. (A magasságát a **c.height** tulajdonság adná meg.) Azt már nem írjuk le, hogy a font mire jó, magatoktól is rájöttök.-)



Beküldendő feladatok

1. Készítsétek el az olimpiai öt karikát azzal az eltéréssel, hogy a körök legyenek kitöltve is (a szegély maradjon) a következő módon: minden kör legyen [színátmenetes](#) mégpedig azzal a 2-5 színnel, amilyen az adott kör és a szomszédja(i) rendelkeznek. Például a piros "karika" piros és zöld színekkel, a zöld viszont zöld, fekete és piros színekkel legyen kitöltve! Minden kör sugara 70 pixel legyen! Minden kör belsejében legyen ott egy használt szín neve úgy, hogy az olvasható legyen! (11110|2)

2. Rajzoljátok meg a csapatlogókat JavaScript kódban egyszerű alakzatok felhasználásával! Az elkészült logót a továbbiakban minden beadott feladatnál felhasználhatjátok. (101000|2)

- A figuratív megjelenésnél ügyeljete a jól felismerhető formák kiválasztására.
- A megfelelő színek használata erősíti az üzenetet. Használjatok legalább három színt a mondanivalótok kiemeléséhez.
- A logók egyik legfontosabb alkotóeleme a felirat, azaz a tipográfia. A tudatos, átgondolt betűhasználat növelheti a logó értékét.
- Legyen méretezhető, ne okozzon gondot a megjelenés a canvas méretének megváltoztatásakor!