JavaScript bevezetés

A kezdet

Kedves Bitfaragók! Kezdődjék hát az első forduló. Ennek az első szakasznak a lényege a JavaScript alapok megismerése. Ha még csak ismerkedtek ezzel a nyelvvel, akkor ehhez most kaphattok iránymutatást. A túra során lesz több gyakorló feladat, amik a saját gyakorlásra vannak, és nem kell őket beküldeni. Végül az oldal alján találhatóak a mostani fordulóra beküldendő feladatok, amelyeknek megoldásához az itt tárgyalt ismeretekre lesz szükség. Természetes akik már ismerősek a JavaScript terén, nyugodtan rátérhetnek egyből a feladatokra.

1. Kockadobást szimuláló program készítése.

Bemenet: 6 fáziskép egy dobókockáról. Felhasználhatók Juhász Tibor-Kiss Zsolt: Programozási ismeretek (MK-4462-3) tankönyv digitális mellékletének képei, ám az alábbi képek letöltésének sincs akadálya:

Kimenet: Az előállított véletlen értéket ábrázoló kép.
Változók: Random szám 1 és 6 között, String típusú tömb 6 elemmel
Algoritmus: Feltöltjük az imgsrc tömböt a képek nevével, majd az előállított véletlenszám segítségével kiválasztjuk azt a tömbelemet, amellyel a HTML dokumentumban megjelenített kép lecserélhető. A sikeres megoldáshoz a DOM modell felépítését kell megismernünk.
Eseménykezelés: Az onclick esemény bekövetkezésekor végrehajtandó utasításokat a changeImage() függvényben rögzítjük.
Az alábbiakban rendszerezzük a megoldáshoz szükséges ismereteket. A platformfüggetlen megoldás elkészítéséhez gyakorlatilag egy böngésző programra, és esetleg egy egyszerűbb editorra van szükség. A lecke mintaprogramjai a böngészőben kipróbálhatók és módosíthatók a forráskód alatti "Próbáld ki! »" gombra kattintás után. Alea iacta est.

Hogyan épül fel egy HTML dokumentum?

Elöszőr gondoljuk át a programozási környezet jellemzőit. A böngészők HTML kódokat jelenítenek meg. A HTML nyelvet azonban még nem tekintjük programnyelvnek, mert csupán a tartalmi elemek megjelenítéséhez szükséges információkat írja le. Vizsgáljuk meg HTML dokumentum szerkezetét. Mindig <html> és a </html> tagek (HTML utasítások) között találhatók a tartalmi elemek. A <head> és a </head> között a dokumentum meta (leíró) információi szerepelnek. Az itt található adatok nem jelennek meg a böngészőben, mivel a dokumentum tulajdonságairól adnak információt a böngészőnek és a robotoknak. A dokumentum tartalmi része a <body> és a </body> tagek között található. Egy HTML dokumentumban szöveg, kép, hang és mozgókép jeleníthető meg. Készítéséhez használhatunk egyszerűbb texteditort (pl. notepad, notepad++), vagy HTML fejlesztő programot (pl. Adobe Dreamweaver, CoffeeCup).

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML dokumentum felépítése</title>
  <meta charset="UTF-8">
</head>

<body>
  <h1>Címsor</h1>

  <p>Bekezdés</p>

  <ul>
    <li>Felsorolás első eleme</li>
    <li>Felsorolás második eleme</li>
    <li>Felsorolás Harmadik eleme</li>
   </ul>

  <p>Hivatkozás:<br />
    <a href="https://mik.uni-pannon.hu/index.php/hu/bbb2021" >Ugrás a
verseny felhíváshoz</a>
  </p>

  <p>Kép:<br /><br />
  <img id="myimage" src="https://mik.uni-pannon.hu/images/cimer120.png"
alt="Kocka" width="100" height="100" />
  </p>
</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

I. A fenti minta tanulmányozása után készítsetek egy bemutatkozó HTML dokumentumot, amelyben megjelenik a csapat neve, az iskolájának neve, a felkészítő tanár neve, valamint külön fényképek minden csapattagól, amik felett a nevek is megjelennek. A honlapon jelenjen meg az is, hogy mikor készítettétek el a feladatot (dátum).

II. Keressetek pár olyan oldalt, amik programozási segédleteket, tutorialokat tartalmaznak. Készítsetek egy HTML dokumentumot, amely ezek linkjeit tartalmazza.

A HTML oldalon a <form> és a </form> tagek által közrefogott űrlapról tudunk adatokat beolvasni. Az adatok bekérésének leggyakrabban használt utasítása az <input> tag.

```
<!DOCTYPE html>
<html>
<head>
  <title>Űrlap</title>
  <meta charset="UTF-8">
</head>

<body>
  <h1>Űrlap demo</h1>

  <form action="./html_step02.php" method="POST">
    Vezetéknév: <input type="text" name="veznev" value="Chat"><br>
    Keresztnév: <input type="text" name="utonev" value="Elek"><br>
    <input type="submit" value="Elküld">
  </form>

  <p>Kattints az elküld gombra, és az űrlapadatokat a szerver
html_step02.php oldala feldolgozza.</p>
</body>
</html>
```

<Experts only!>
Pusztán az ügyfél-kiszolgáló architektúra megértésének kedvéért, szánjunk pár percet a szerver oldali adatfeldolgozást végző PHP kód áttekintésére is. Figyeljük meg, hogy az űrlap <form> tagja rendelkezik egy **action** attributummal, amelyben megadhatjuk, hogy milyen kódot hajtson végre a szerver az "Elküld" gomb lenyomását követően. Az űrlapmezők adatait a $_POST tömbben fogadja a szerver.

```
<!DOCTYPE html>
<html>
<head>
  <title>Haladóknak - Űrlapfeldolgozás</title>
  <meta charset="UTF-8">
</head>

<body>

<?php
  print "<h1>A szerver az alábbi adatokat kapta:</h1>";
  print var_dump($_POST);
  print "<hr />";
  print "<h1>Kiszolgáló információk:</h1>";
  print "IP cím: ".$_SERVER['SERVER_ADDR']."<br />";
  print "Név: ".$_SERVER['SERVER_NAME']."<br />";
  print "Szoftver: ".$_SERVER['SERVER_SOFTWARE']."<br />";
  print "<hr />";
  print "<h1>Ügyfél információk:</h1>";
  print "IP cím: ".getenv('REMOTE_ADDR')."<br />" ;
  print "Név: ".gethostbyaddr($_SERVER['REMOTE_ADDR'])."<br />";
  print "Szoftver: ".$_SERVER['HTTP_USER_AGENT']."<br />";
  print "<br />Hivatkozás: ".$_SERVER['HTTP_REFERER']."<br />";
?>

</body>
</html>
```

A fenti kód a szerveren fut, a böngésző csak a futtatás eredményét kapja vissza. Ugye észrevetted, hogy itt a <?php és ?> jelölések között jelentek meg a PHP utasítások. Ha szeretnéd kipróbáni a szerver oldali programozást, akkor telepíts a számítógépedre egy WAMP szervert vagy regisztrálj egy ingyenes webtárhelyet, ám a JavaScript programozáshoz erre egyenlőre nem lesz szükséged.
</Experts only!>

JavaScript utasítások a HTML dokumentumban

Ha a HTML kódban számításokat akarunk végezni, vagy néhány feltétel teljesülését szeretnénk megvizsgálni, vagy esetleg utasításokat többször végrahajtani, akkor bizony adat- és vezérlőszerkezetekre lesz szükségünk. Nos, a HTML ezen hiányosságát pótolhatjuk például JavaScript kódok beágyazásával. Ilyenkor a <script> és a </script> tagek közé írjuk a JavaScript utasításokat, melyeket a böngésző program fog értelmezni és végrehajtani. Az alábbi program például az aktuális dátum megjelenítésére szolgál. A megjelenítést egy létrehozott függvény végzi el, ami akkor kerül meghívásra, amikor valaki a gombra kattint. A függvény megkeresi azonosító alapján az oldal egy bekezdését, és azt módosítja. További infó a dátumokról itt.

```
<!DOCTYPE html>
<html>
<head>
   <title>Utasítások</title>
 <meta charset="UTF-8">
<script>
  function displayDate() {
    alert("Hello...\n\nMost a böngésző értelmezi a JavaScript kódot.");
    var d=new Date();
    document.getElementById("demo").innerHTML=d.toLocaleDateString();
  }
</script>

</head>
<body>

<h1>Első JavaScript kódom</h1>
<p id="demo">Ebbe a bekezdésbe kerül a dátum, ha a gombra kattintasz</p>

<button type="button" onclick="displayDate()">Dátumkiírás</button>

</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

III. Készítsetek egy oldalt, ami betöltéskor megjeleníti az akkori aktuális időt. Legyen az oldalon egy gomb, amit megnyomva az idő frissül.

IV. Készítsetek egy oldalt, ahol egy gomb lenyomására helyette megjelenik egy karácsonyfa képe.

JavaScript változók használata

A változót névvel azonosított memóriaterületnek tekintjük, különböző értékek tárolásának illetve 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 értelmező végzi el. Az eredmény a kifejezésben használt változók és a kijelölt műveletek jellegétől függően leggyakrabban szám, szöveg vagy logikai típusú.

```
<!DOCTYPE html>
<html>
<head>
  <title>Változók</title>
  <meta charset="UTF-8">
  <script>
    // Szám típusú változók
    var a;
    var a = 3;
    var b = 4;

    // Értékadó kifejezések
     var osszeg = a+b;
   var c = Math.sqrt(a*a+b*b);

    // Szöveg típusú változó
    var s = "Derékszögű háromszög oldalai:";
  </script>
</head>

<body>
  <h1>JavaScript változók használata</h1>
  <script>
    document.write(a + " + " + b + " = " + osszeg + "<br />")
    document.write("<h2>"+s+"</h2>")
    document.write("Befogók: "+a + ", " + b + " => Átfogó: " + c + "<br
/>")
  </script>

</body>
</html>
```

Próbáld ki!

Ha megegyező típusú adatok sorozatát szeretnénk feldolgozni (pl. osztályzatok, hőmérséklet, bevételek, kiadások), akkor tömbösíthetünk, vagyis egy közös azonosítóval hivatkozhatunk az adatsorozatra, miközben annak elemeit sorszámmal jelöljük. Tömb deklarálása az Array() kulcsszóval történik.

```
<!DOCTYPE html>
<html>
<head>
  <title>Változók</title>
  <meta charset="UTF-8">
  <script>
    // Tömb definiálása (Az indexelés mindig 0-tól kezdődik!)
    var autok = new Array();
    autok[0]="Mercedes";
    autok[1]="Lexus";
    autok[2]="Nissan";

    var jegyek = new Array(3,4,5,4,2,5);
  </script>
</head>

<body>
  <h1>JavaScript tömb használata</h1>
  <script>
    document.write("Autók: "+autok[0] + ", " + autok[1] + ", " + autok[2] +
"<br />");
  </script>

  <h1>JavaScript tömbelemek feldolgozása</h1>
  <p>Bővebben a vezérlőszerkezeteknél...</p>
  <script>
    var osszeg=0;
    for(i=0;i<jegyek.length;i++) {
      document.write(jegyek[i] + ", ");
      osszeg+=jegyek[i];
    }
    document.write("<h2>Átlag:
"+(osszeg/jegyek.length).toFixed(2)+"</h2>");
  </script>
  </body>
</html>
```

Próbáld ki!

Billentyűzetről a legegyszerűbben a prompt() metódussal kérhetünk be adatot:

```
<!DOCTYPE html>
<html>
<!-- http://www.w3schools.com/js/tryit.asp?filename=tryjs_prompt -->
<head>
<title>Változók</title>
<meta charset="UTF-8">
<script>
  function myFunction() {
    var x;

    var person=prompt("Kérem írja be a nevét","Harry Potter");

    if (person!=null)  {
      x="Hello " + person + "! Hogy vagy?";
      document.getElementById("demo").innerHTML=x;
    }
  }
</script>
</head>
<body>

<h1>Prompt() metódus bemutatása!</h1>

<button onclick="myFunction()">Adatbeolvasás</button>

<p id="demo"></p>
</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

V. Készítsetek egy programot, amely bekér 5 egész értéket, és egy tömbben tárolja őket. A program ezután jelenítse mindegyik elem négyzetét.

VI. Kérjetek be egy lebegőpontos számot, majd jelenítsétek meg az egészre kerekített értékét.

JavaScript vezérlőszerkezetek

Meghatározott logikai feltételek teljesülése esetén a vezérlőszerkezetekkel tudjuk szabályozni az utasítások végrehajtásának sorrendjét. A struktúrált programban alapvetően három vezérlőszerkezet használatos: szekvencia, szelekció, iteráció. Figyeljük meg a mintapéldákban a vezérlőszerkezetek egymásba ágyazódását!

Szekvencia

Szekvencia (utasításblokk) esetén az utasítások a programba írt sorrendjük szerint hajtódnak végre.

```
<!DOCTYPE html>
<html>
<head>
<title>Vezérlőszerkezetek</title>
<meta charset="UTF-8">
<script>
  // Szám típusú változo
  var a = 5;
</script>
</head>

<body>
<h1>Számítások:</h1>
<script>
  document.write("<p>"+ a + " cm oldalú négyzet területe: " + a*a +
"</p>");
  document.write("<p>"+ a + " cm oldalú kocka felszíne: " + 6*a*a +
"</p>");

  var r=a;
  var korterulet=r*r*Math.PI;
  document.write("<p>"+ r + " cm sugarú kör területe: " +
korterulet.toFixed(4) + "</p>");
  var gombfelszin=4*r*r*Math.PI;
  document.write("<p>"+ r + " cm sugarú gömb felszíne: " +
gombfelszin.toFixed(4) + "</p>");
</script>

</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

VII. A program oldjon meg másodfokú egyenletet: Kérjen be 3 számot (a, b, és c), majd a megoldóképletet alkalmazva adja meg mindkét eredményt (feltehetjük, hogy van neki).

VIII. Egy egyszerűsített fizikai rendszerben a testekre csak a gravitáció hat, melynek mértéke 9.81 m/s^2. Egy ilyen rendszerben egy eldobott labda röppályálya könnyen modellezhető. Készítsetek egy programot, ami bekéri a labda eldobását jelző ferde hajtás erősségét (v_0, pozitív szám) és szögét (α, 0 és 90 fok között), és kiszámolja a dobás maximális magasságát, illetve távolságát.

Ferde hajítás segítség

Figyelem, Javascript-ben a sin és cos függvények radiánban várják a szöget, nem fokban!

Szelekció

Szelekcióhoz (elágazás) már olyan utasításokat használunk, amelyek valamilyen logikai kifejezés értékétől függően eltérő utasításblokkot hajtanak végre. Megvalósításához használhatjuk az if (logikai kifejezés){ } utasítást.

```
<!DOCTYPE html>
<html>
<head>
<title>Vezérlőszerkezetek</title>
<meta charset="UTF-8">
<script>
  // Szám típusú változók
  var a = 5;
  var b = 12;
</script>
</head>

<body>
<script>
  document.write("<h1>"+ a + " vagy " + b + " a nagyobb?" + "</h1><br />");
  if (a>b) {
    document.write(a+" nagyobb mint " + b);
  }
  else if (b>a) {
    document.write(b+" nagyobb mint " + a);
  }
  else {
    document.write(a+ " egyenlő "+ b + "<br />");
  }
</script>

</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

IX. Készítsetek egy oldal, ami bekér egy időpontot (óra, perc) 24 órás formátumban, és kiírja, hogy az az időpont munkaidőben van-e. Vegyük azt, hogy a munkaidő 7:30-tól 16:15-ig tart.

X. Készítsetek egy programot, amely bekér 5 számot, és kiírja hogy melyik a legnagyobb.

Iteráció

Az iteráció (ismétlődés, ciklus) teszi lehetővé bizonyos feltételek teljesülése esetén egy utasításblokk többszöri végrehajtását. Ilyenkor a ciklusmagban használt változók értéke feladattól függően folyamatosan változhat. Az while (logikai kifejezés){ } elöltesztelő ciklusnál az ismétlési feltételeket a ciklus mag előtt allítjuk be, míg a do { } while (logikai kifejezés) hátultesztelő ciklus esetén a ciklusmag utasításblokkját követően. Létezik még az úgynevezett növeknényes ciklusszervező utasítás a for (ismétlések) { }. Itt az ismétlések tényleges számát már a ciklus indulásakor meg kell adnunk. Figyeljük meg az alábbi mintaprogramban az iterációk felépítését.

```
<!DOCTYPE html>
<html>
<head>
<title>Vezérlőszerkezetek</title>
<meta charset="UTF-8">
</head>

<body>
<script>
  var a=1;
  var b=12;
  document.write("<h1>Egész számok "+a+" és "+b+" között</h1>");
  for (i=a;i<=b;i++) {
    document.write(i + ", ");
  }
  document.write("<br />");

  document.write("<h1>Páros számok "+a+" és "+b+" között</h1>");
  var j=a;
  while(j<=b) {
    if(j%2==0) {
      document.write(j + ", ");
    }
    j++;
  }
  document.write("<br />");

  document.write("<h1>Páratlan számok "+a+" és "+b+" között</h1>");
  var j=a;
  do {
    if(j%2!=0) {
      document.write(j + ", ");
    }
    j++;
  } while(j<=b)
  document.write("<br />");

</script>

</body>
</html>
```

Próbáld ki!

Gondoltam egy számot: A program kigondol egy számot egy és tíz között, majd megkérdezi tőlünk, hogy vajon melyikre gondolt. Ezt addig teszi, amíg ki nem találjuk. A probléma megoldására while ciklust és if elágazást használunk. Figyeljétek meg a véletlen egész értékek generálásának módját. Az eredmény lehet 1 is és 10 is.

```
<!DOCTYPE html>
<html>
<head>
<title>Vezérlőszerkezetek</title>
<meta charset="UTF-8">
<script>
  function myFunction() {
    var numberComputer = Math.floor((Math.random()*10)+1);
    var num = prompt("Gondoltam egy számra egy és tíz között. Találd ki!");
    var ok = true;
    while(ok) {
      if(isNaN(num)){
        num = prompt("Csak számot adhatsz meg és legyen egy és tíz
között!");
      } else if(num < numberComputer) {
        num = prompt("Nagyobbra gondoltam!");
      } else if(num > numberComputer) {
        num = prompt("Kisebbre gondoltam!");
      } else if(num == numberComputer) {
        alert("Igen, erre a számra gondoltam: " + numberComputer);
        ok = false;
      }
    }
  }
</script>
</head>

<body>

<h1>Gondoltam egy számot!</h1>
<button type="button" onclick="myFunction()">Találd ki!</button>
<p id="demo"></p>

</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

XI. Készítsetek egy programot, amely egy gombra kattintva bekér egy számot (ezt akármennyiszer megcsinálhatjuk). A program mindig tartsa számon, hogy az indulás óta mennyi 5-tel osztható számot írtunk be, és ez mindig legyen látható az oldalon.

XII. Készítsetek egy programot, amely egy 20 elemű tömböt feltölt véletlen egész számokkal, 0 és 100 között. A generálás után jelenítsétek meg a generált számokat, valamint azok összegét is.

XIII. Készítsetek egy oldalt, ami egész számokat kér be. A bekérés akkor álljon le, ha 0-t írunk be. A bekérés során az oldalon jelenjenek meg a bekért számok (mindig frissülve az új bekéréskor).

JavaScript függvények és az eseménykezelés

Függvényhívások

A függvény névvel azonosított utasításcsoport. Általában akkor alkalmazzuk, amikor ugyanazokat a műveleteket kell a program különböző részein elvégezni, esetleg más-más adattal. A function kulcsszó után az általunk magadott néven lesznek elérhetőek az utasítás zárójelek { } között megadott utasítások. Ha a függvény utasításainak adatot szeretnénk átadni, akkor a függvény neve után zárójelben megadott úgynevezett attribútumokkal tehetjük meg. A függvény hívása nevének és attribútumainak megadásával történik.

```
<!DOCTYPE html>
<html>
<head>
<title>Függvények és események</title>
<meta charset="UTF-8">
<script>
  function myexplorer() {
    txt = "<p>Böngésző kódnév: " + navigator.appCodeName + "</p>";
    txt+= "<p>Böngésző név:  " + navigator.appName + "</p>";
    txt+= "<p>Böngésző verzió: " + navigator.appVersion + "</p>";
    txt+= "<p>Süti engedélyezve: " + navigator.cookieEnabled + "</p>";
    txt+= "<p>Platform: " + navigator.platform + "</p>";
    txt+= "<p>User-agent header: " + navigator.userAgent + "</p>";
    txt+= "<p>User-agent nyelv: " + navigator.systemLanguage + "</p>";
    document.getElementById("example").innerHTML=txt;
  }
</script>
</head>

<body>

  <h1>Függvény hívások</h1>
  <p id="example">Ebbe a bekezdésbe kerülnek a böngésző adatai</p>

  <button type="button" onclick="myexplorer()">Böngészőm adatai</button>

</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

XIV. Készítsetek egy függvényt, amely megkap egy tömböt, és visszaadja az elemek összegét. Az oldal kérjen be 5 számot, majd a függvény segítségével számítsa ki és jelenítse mg az összegüket.

XV. Készítsetek egy függvény, ami megkap egy szöveget, és egyszerű eltolásos kódolással (Caesar kóddal) kódolja: minden karaktert eltol eggyel (a-ből b lesz, b-ből c, c-ből d, stb., végül a z-ből a). Csak az angol abc kis betűivel kell foglalkozni. Az oldal kérjen be egy szöveget, és jelenítse meg a kódolt verzióját.

XVI. Az előző feladatot tovább fejlesztve legyen egy függvény, ami a szöveg mellé az eltolás mértékét is megkapja, és minden karaktert ennyivel tol el (pl. 3 esetén a-ből d lesz, b-ből e, y-ból b, stb). Mellette legyen egy visszafeltő függvény is, ami ugyanezt csinálja, csak a fordított irányba. Az oldal kérje be a szöveget, az eltolás mértékét, és hogy kódolás vagy dekódolás kell, hajtsa végre a műveletet, megjelenítve az eredményt.

Eseménykezelés

A program működése közben a kezelőfelületről illetve a kommunikációs felületről érkező események üzeneteit folyamatosan dolgozza fel, így a külső (egérről vagy billentyűzetről érkező) és a belső (timertől vagy más működő folyamatoktól érkező) események befolyásolják a futását.
A külső események érzékelésekor az operációs rendszer üzenetet küld annak az ablaknak, amelyen az esemény bekövetkezett. Az alkalmazás fogadja az üzenetet, és végrehajtja az eseményhez rendelt utasításokat. Ilyen események lehetnek az egérkattintás, az egérkurzor valamelyik dokumentumelem feletti állapota stb.

```
<!DOCTYPE html>
<html>
<head>
<title>Függvények és események</title>
<meta charset="UTF-8">
<script>
  function Hatter(color){
    alert("Háttérszín:
"+document.getElementById("myPage").style.backgroundColor+" -> "+color);
    document.getElementById("myPage").style.background=color;
  }
</script>
</head>

<body id="myPage" style="background:lightyellow;" >
<h1>Háttérszíncsere egérkatintásra</h1>
<table style="text-align:center;padding:5px;">
  <tr>
    <td><input TYPE="button" value="Piros háttér"
onclick="Hatter('red')"></td>
    <td><input TYPE="button" value="Zöld háttér"
onclick="Hatter('green')"></td>
    <td><input TYPE="button" value="Kék háttér"
onclick="Hatter('blue')"></td>
    <td><input TYPE="button" value="Eredeti háttér"
onclick="Hatter('lightyellow')"></td>
  </tr>
</table>

</body>
</html>
```

Próbáld ki!

Szintén külső események vezérlik az űrlapmezők kitöltését, vagy az alábbi **kő-papír-olló** játékot is.

```
<!DOCTYPE html>
<html>

<head>
<title>Ko-papír-olló játék</title>
<meta charset="UTF-8">
<script>
  function myfunction(){
    alert("Ko, papír, olló!");
    document.getElementById("result").style.visibility="hidden";
    document.getElementById("demo").style.visibility="visible";
    document.getElementById("hi").innerHTML="<h1>Hello " +
document.getElementById("veznev").value+" " +
document.getElementById("utonev").value+"!</h1><p>Ko, papír
vagy olló?</p>";
  }
  function kpofunction(userTip){
    var compTip=Math.floor((Math.random()*3)+1);
    document.getElementById("userT").src="./img/KPO"+userTip+".jpg";
    if (userTip==compTip){
      document.getElementById("winner").innerHTML=" = ";
    } else{
      if ((3+userTip-compTip)%3==1) // Át lehet gondolni, hogy miért jó
ez így
        document.getElementById("winner").innerHTML=" > ";
      else
        document.getElementById("winner").innerHTML=" < ";
      }
    document.getElementById("compT").src="./img/KPO"+compTip+".jpg";
    document.getElementById("result").style.visibility="visible";
  }
</script>
</head>

<body>
<h1>JavaScript ko-papír-olló</h1>

<form action="javascript:myfunction()">
  Vezetéknév: <input type="text" id="veznev" name="veznev"
value="Chat"><br>
  Keresztnév: <input type="text" id="utonev" name="utonev"
value="Elek"><br>
  <input type="submit" value="Feldolgoz">
</form>

<p>Kattints az feldolgoz gombra.</p>

<div id="demo" style="visibility:hidden;">
<p id="hi"></p>
<p id="tip">
<img id="kpo3" src="./img/KPO3.jpg" alt="Ko" onclick="kpofunction(3)"/>
<img id="kpo1" src="./img/KPO1.jpg" alt="Papír" onclick="kpofunction(1)"/>
<img id="kpo2" src="./img/KPO2.jpg" alt="Olló" onclick="kpofunction(2)"/>
</p>
</div>
```

```
<div id="result" style="visibility:hidden;">
<table>
  <tr>
    <td>Játékos:<br />  <img id="userT" alt="Játékos"
src="./img/KPO1.jpg" /></td>
    <td style="width:70px;text-align:center;"><h1 id="winner">
 </h1></td>
    <td>Computer:<br /> <img id="compT" alt="Computer"
src="./img/KPO1.jpg" /></td>
  </tr>
</table>
</div>

</body>
</html>
```

Ezt a belső időzítést a továbbiakban természetesen más rendszeresen ismétlődő tevékenységek elvégzéséhez is használhatjuk.

```
<!DOCTYPE html>
<html>
<head>
<title>Függvények és események</title>
<meta charset="UTF-8">
<script>
  var myVar=setInterval(function(){myTimer()},1000);

  function myTimer() {
    var d=new Date();
    var t=d.toLocaleTimeString();
    document.getElementById("demo").innerHTML=t;
  }
</script>
</head>

<body>

<h1>A pontos idő: <span id="demo" style="color:#00A000;"></span></h1>

</body>
</html>
```

Próbáld ki!

Gyakorló feladatok:

XVII. Készítsetek egy honlapot, ami megjelenít egy rövid mesét. Az oldal kérjen be a felhasználótól rövid szövegeket. A bekérés után egy gombra kattintva jelenítse meg a szövegeket. Egyszerre csak egy jelenjen meg, majd adott időközönként váltson a következőre.

XVIII. Készítsetek oldalt, ami egy gyorsasági gépelési játékot valósít meg. Az oldal a kezdés gombra kattintva adjon 1-2 másodperc felkészülési időt, majd jelenítsen meg egy véletlenszerű szót (legyen valami halmaz, ahonnét válogathat). A felhasználónak minél gyorsabban be kell gépelnie a szót, majd elküldeni (enter-rel vagy gombnyomással). Az oldal ezután írja ki, hogy helyes volt-e a szöveg, és mennyi időt vett igénybe a begépelés.

XIX. Készítsetek egy visszaszámláló honlapot. Lehessen megadni hogy hány másodperces legyen a visszaszámolás, majd elindítani. Az oldal jelezze a hátralévő másodperceket, és ha eléri a 0-t, akkor dobjon fel erről egy üzenetet.

A kockadobás feladat lehetséges megoldásai

A leckében tárgyaltak összefoglalásaként nézzük meg az eredeti feladat néhány lehetséges megoldását.

Kockadobás szelekció használatával

```
<!DOCTYPE html>
<html>

<head>
<title>Kockadobás</title>
<meta charset="UTF-8">
</head>

<body>
<script>
  var rnd;
  function changeImage()
  {
    rnd=Math.floor((Math.random()*6)+1);
    element=document.getElementById('myimage')
    if (rnd==1) {
      element.src="./img/Kocka1.png";
    }
    else if (rnd==2) {
      element.src="./img/Kocka2.png";
    }
    else if (rnd==3) {
      element.src="./img/Kocka3.png";
    }
    else if (rnd==4) {
      element.src="./img/Kocka4.png";
    }
    else if (rnd==5) {
      element.src="./img/Kocka5.png";
    }
    else {
      element.src="./img/Kocka6.png";
    }
  }
</script>

<img id="myimage" onclick="changeImage()" src="./img/Kocka6.png"
alt="Kocka" width="100" height="100">

<p><a href="#" onclick="changeImage()">Kattints a kockára!</a></p>

</body>
</html>
```

Kockadobás tömb használatával

```
<!DOCTYPE html>
<html>

<head>
<title>Kockadobás</title>
<meta charset="UTF-8">
</head>

<body>
<script>
  var rnd;
  imgsrc = new Array(6);
  imgsrc[0] = "./img/Kocka1.png";
  imgsrc[1] = "./img/Kocka2.png";
  imgsrc[2] = "./img/Kocka3.png";
  imgsrc[3] = "./img/Kocka4.png";
  imgsrc[4] = "./img/Kocka5.png";
  imgsrc[5] = "./img/Kocka6.png";
  function changeImage() {
    document.getElementById('myimage').setAttribute("src",
imgsrc[Math.floor(Math.random()*6)]);
  }
</script>

<img id="myimage" onclick="changeImage()" src="./img/Kocka6.png"
alt="Kocka" width="100" height="100">

<p><a href="#" onclick="changeImage()">Kattints a kockára!</a></p>

</body>
</html>
```

Kockadobás iteráció használatával

```
<!DOCTYPE html>
<html>
<head>
<title>Kockadobás</title>
<meta charset="UTF-8">
</head>

<body>
<script>
  var rnd;
  imgsrc = new Array(6);
  for (var i=0;i<imgsrc.length;i++) {
    imgsrc[i] = "./img/Kocka"+(i+1)+".png";
  }

  function changeImage() {
    document.getElementById('myimage').setAttribute("src",
imgsrc[Math.floor(Math.random()*6)]);
  }
</script>

<img id="myimage" onclick="changeImage()" src="./img/Kocka6.png"
alt="Kocka" width="100" height="100">
<p><a href="#" onclick="changeImage()">Kattints a kockára!</a></p>
</body>
</html>
```

Egy kis kiegészítés

Egyéb beviteli mezők, és kezelésük

A HTML-ben több lehetőség is rendekezésre áll felhasználói bemenetre. A különböző fajta beviteli mezők között szereplnek szöveget, számot kezelőek, jelölőnégyzetek.
Az alábbi mintaprogram bemutatja egy pár ilyen mezőnek a kezelését javascript segítségével.

```
<!DOCTYPE html>
<html lang="hu">
<head>
<title>Beviteli mezők</title>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<script>
  function adatelemzes(){
    var valasz="";
    valasz+="Üdvözöllek " + document.getElementById("nev").value +
"!<br>\n";
    valasz+=document.getElementById("kor").value + " a legszebb
kor.<br>\n";

    var nyelv_index=document.getElementById("nyelv").selectedIndex;
    var nyelv_azonosito = document.getElementById("nyelv").value;
    var nyelv_szoveg =
document.getElementById("nyelv").options[nyelv_index].text;
    if (nyelv_azonosito=="angol" || nyelv_azonosito=="olasz" ||
nyelv_azonosito=="orosz")
      valasz+="Szép is az " + nyelv_szoveg + " nyelv.<br>\n";
    else
      valasz+="Szép is a " + nyelv_szoveg + " nyelv.<br>\n";
    document.getElementById("visszajelzes").innerHTML=valasz;

    var progok=document.getElementsByName("prog")
    if (progok[0].checked){
      document.getElementById("progvalasz").innerHTML="Nagyon helyes!";
      document.getElementById("progvalasz").setAttribute("color",
"#00ff00");
    }
    else{
      document.getElementById("progvalasz").innerHTML="De akkor miért vagy
ezen az oldalon??";
      document.getElementById("progvalasz").setAttribute("color",
"#ff0000");
    }
  }

  function proba_valtozott(){
    var bejelolve=document.getElementById("proba").checked;
    if (bejelolve)
      document.getElementById("probaszoveg").setAttribute("color",
"#2222cc");
    else
      document.getElementById("probaszoveg").setAttribute("color",
"#cc2266");
  }
</script>
</head>
```

```
<body>
<h1>Beviteli mezők használata</h1>


Kérlek add meg a neved:
<input type="text" id="nev"><br>

Életkorod:
<input type="number" id="kor"><br>

Válaszd ki a kedvenc idegen nyelved:
<select id="nyelv">
  <option value="angol">Angol</option>
  <option value="francia">Francia</option>
  <option value="nemet">Német</option>
  <option value="olasz">Olasz</option>
  <option value="orosz">Orosz</option>
  <option value="spanyol">Spanyol</option>
</select><br>

Szereted a programozást?
<input type="radio" name="prog" value="igen"> Igen
<input type="radio" name="prog" value="nem"> Nem
<font id="progvalasz"></font><br>

Ezt is próbáld ki!
<input type="checkbox" id="proba" onchange="proba_valtozott();"> <font
id="probaszoveg">Próba!!!</font><br>

<button onclick="adatelemzes();">Hagy menjen!!</button>

<br><br><br>
<p id="visszajelzes"></p>
<br><br><br>

</body>
</html>
```

Próbáld ki!

Kedves Bakonyi Bitfaragók!

Elérkeztünk az első kirándulásunk végére. Nagy utat tettünk meg a bitek birodalmában, de reméljük a bemutatott minták felhasználásával már érdekes megoldásokat tudtok adni az első forduló feladványaira. Tökéletes program ugyan nem létezik, de nagyon jó megközelítéssel érhetünk el egy kifogástalannak tartott állapotot.

Fontos: A feladatokat egyben kell beküldeni, minden fájlt egy tömörített állományba csomagolva. Legyen a fájlok között egy index.html, amit betöltve egy menü jön be, ahonnan kattintással el lehet navigálni a két feladatra külön-külön. Az egyes feladatok oldalán is jelenjen meg egy hasonló menü, hogy egyszerűen át lehessen térni a másik feladatra.

Beküldhető feladatok:

1) Készítsetek a csapatotoknak egy weboldalt.
   a) Az oldalon jelenjenek meg a csapat adatai (tagok nevei, felkészítő tanár(ok) neve(i), iskola, valamint egy-egy fénykép).
   b) Az oldalon legyen egy óra is, ami mindig az aktuális időt mutatja.
   c) Az oldalon legyenek extra képek is, amik alapból nem látszódnak (lehet a csapatról, de lehet tájkép is). Legyen egy gomb, aminek a lenyomására megjelenik az első kép, majd ismételt lenyomására a következő kép, stb. Ha véget ért a "kivetítés", akkor a következő gombnyomás kezdje elölről.
   d) Minden csapattag nevéhez legyen egy gomb, amire rákattintva meg lehet tekinteni az illetőről egy rövid bemutatkozó szöveget. A gomb ismételt megnyomására a szöveg tűnjön el.

100 0000$|_2$ pont

2) Készítsetek egy film ajánló alkalmazást
   a) A honlap valamilyen módon tároljon filmeket (a név és a megjelenési év a fontos), valamint ezekhez tudja azt is, hogy mely kategóriákba tartoznak (pl. sci-fi, fantasztikus, történelmi, akció, dráma, stb.).
   b) Az oldalon lehessen kategóriákat kiválasztani két csoportba, hogy ez alapján lehessen keresni. Az egyikbe a szimpatikus kategóriákat várjuk, amiket a felhasználó szeretne látni, a másikba a nem kívánatosakat, amiket el szeretne kerülni. A kiválasztás és összegyűjtés módja mindegy, a lényeg, hogy bármelyik csoportba bármennyi kategória mehessen. Például, ha olyan filmet keresek, amiben legyen akció és dráma, de ne legyen se fantasztikus, se sci-fi, akkor az "akció" és "dráma" kategóriákat választom ki a szimpatikus csoportba, és a "fantasztikus" és "sci-fi" kategóriákat a nem kívánatos csoportba.
   c) Ha kiválasztottuk a kategóriákat, akkor egy gomb lenyomására az oldal jelenítse meg az összes olyan filmet az általa ismertek közül, ami megfelel a feltételeknek.

1000 0000$|_2$ pont