WEBFEJLESZTÉS 2. – SÜTIK, HTML5 JAVASCRIPT APIK FELHASZNÁLÓI OBJEKTUMOK

Horváth Győző

Egyetemi adjunktus

1117 Budapest,
Pázmány Péter sétány 1/C, 2.420
Tel: (1) 372-2500/1816

Ismétlés

Ismétlés

- JavaScript nyelvi elemei
- JavaScript és HTML kapcsolata
- Eseménykezelés részletei
- JavaScript beépített objektumok
- DOM és HTML DOM: űrlapok, képek, táblázatok
- BOM
- Névtelen függvények, reguláris kifejezések
- Stílusok, animációk

Tárolás

Tárolás

- Változók
- HTML 5 Storage
  - localStorage
  - sessionStorage
- Süti

localStorage

- Kulcs-érték párok gyűjteménye
  - Objektum

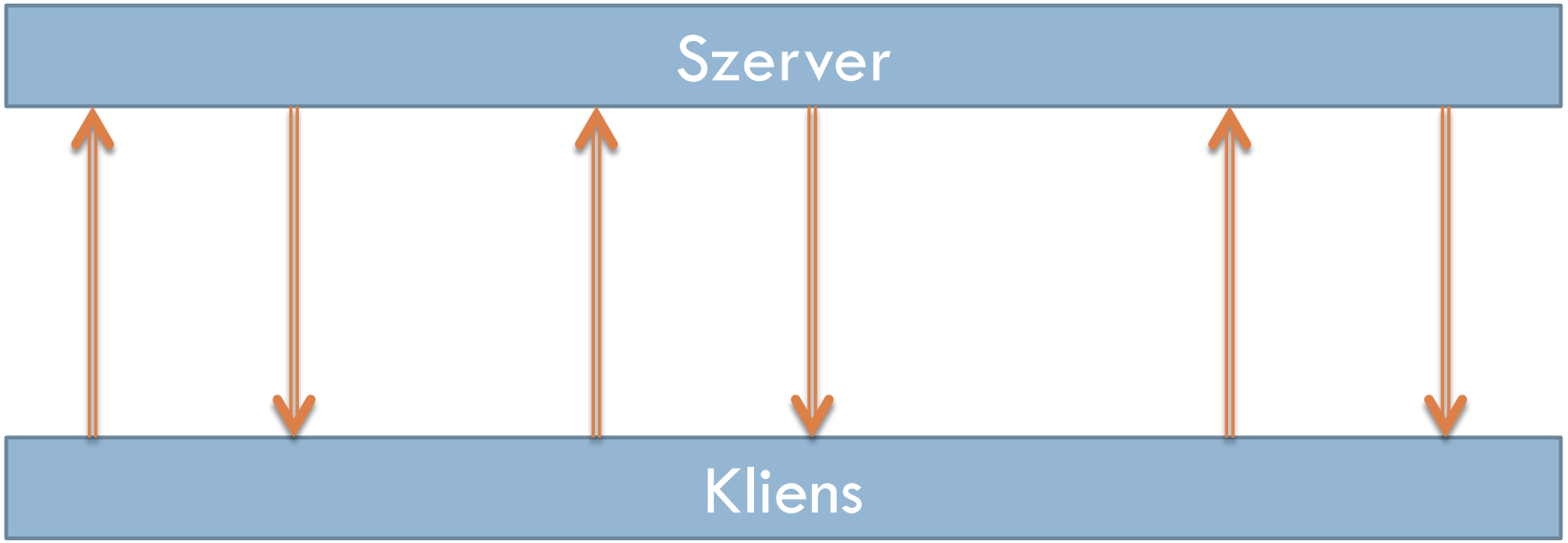
```
//Tárolás
localStorage.alma = 'piros';
localStorage['bármi'] = 42;

//Elővétel (akár újratöltés után)
console.log(localStorage.alma);
console.log(localStorage['bármi']);
```

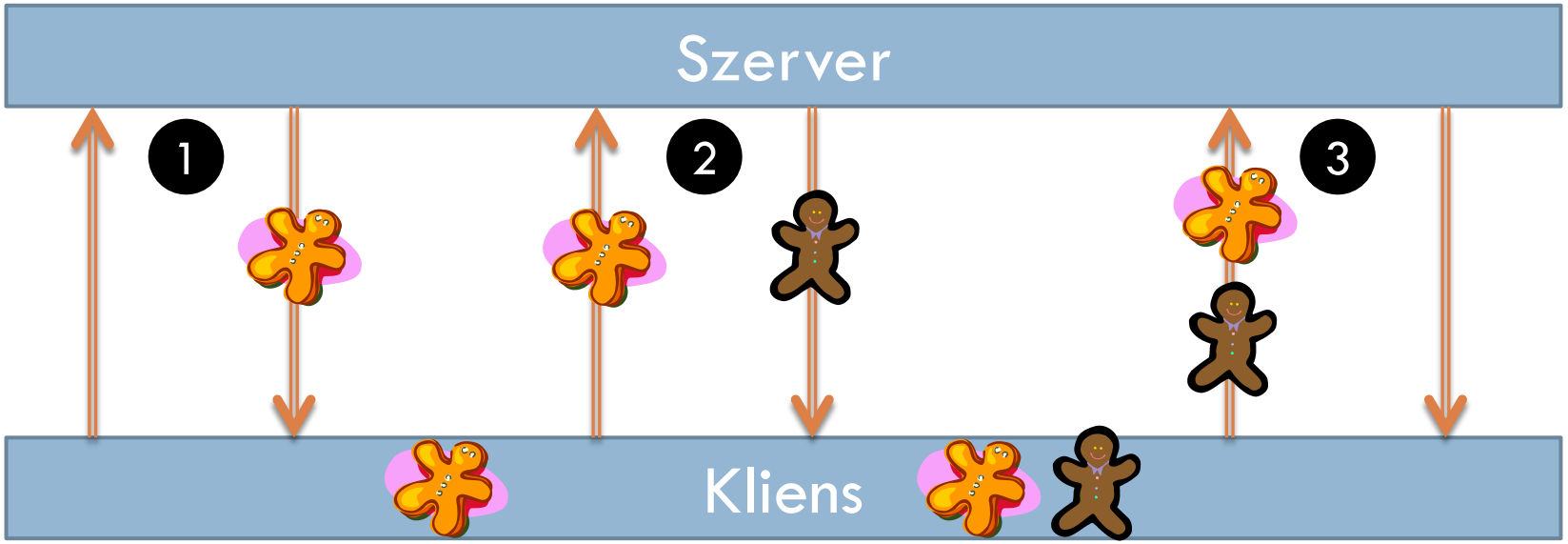
Sütik

- HTTP protokoll állapotmentes
- Két kérés közötti állapot tárolására találták ki
- Böngészőbeli megoldás
- Szöveges információ tárolása
- HTTP protokoll része
  - szervertől érkező válaszban lehet beállítani
  - kérésnél a megfelelő sütik elküldésre kerülnek

Kliens-szerver kérés-válasz



Kliens-szerver kérés-válasz



HTTP protokoll: süti

- Válasz (szerver→kliens)

```
Set-Cookie: név=érték[; expires=dátum][; domain=domain][; path=path][; secure]
```

- Kérés (kliens→szerver)

```
Cookie: név1=érték1; név2=érték2; név3=érték3
```

- Példa süti beállítására

```
Set-Cookie: alma=piros
```

HTTP protokoll

- expires
  - a süti lejáratának dátuma
  - nincs megadva → a session végéig

```
Set-Cookie: alma=piros; expires=Tue, 19 Mar 2013 15:23:07 GMT
```

- max-age
  - a süti élettartama másodpercekben

```
Set-Cookie: alma=piros; max-age=100000
```

HTTP protokoll

- domain
  - melyik oldalhoz tartozik a süti
  - alapértelmezetten: az adott oldal domain-je
  - ki lehet szélesíteni a domain-ek körét
  - más oldalhoz nem lehet megadni path
- path
  - az URL útvonal részét lehet meghatározni
  - balról kezdve nézi az egyezőséget
  - példa: /gyak, /gyakorlat is jó lesz

```
Set-Cookie: alma=piros; domain=webprogramozas.inf.elte.hu
Set-Cookie: alma=piros; domain=elte.hu
Set-Cookie: alma=piros; path=/gyak
```

HTTP protokoll

- secure
  - SSL-es titkosítás

```
Set-Cookie: alma=piros; secure
```

- HttpOnly
  - JavaScriptből nem beállítható, lekérdezhető
  - biztonsági szempontok

```
Set-Cookie: alma=piros; HttpOnly
```

JavaScript: sütik

- document.cookie
- Beállítás

```
document.cookie = 'név=érték[; expires=dátum][; domain=domain]'+
                  '[; path=path][; secure]';
document.cookie = 'alma=piros';
document.cookie = 'korte=sarga; expires=Wed, 20 Mar 2013 15:23:07 GMT';
```

- Lekérdezés

```
document.cookie

console.log(document.cookie);
//"alma=piros; korte=sarga"
```

Segédfüggvények

- Süti beállítása (nem teljes)

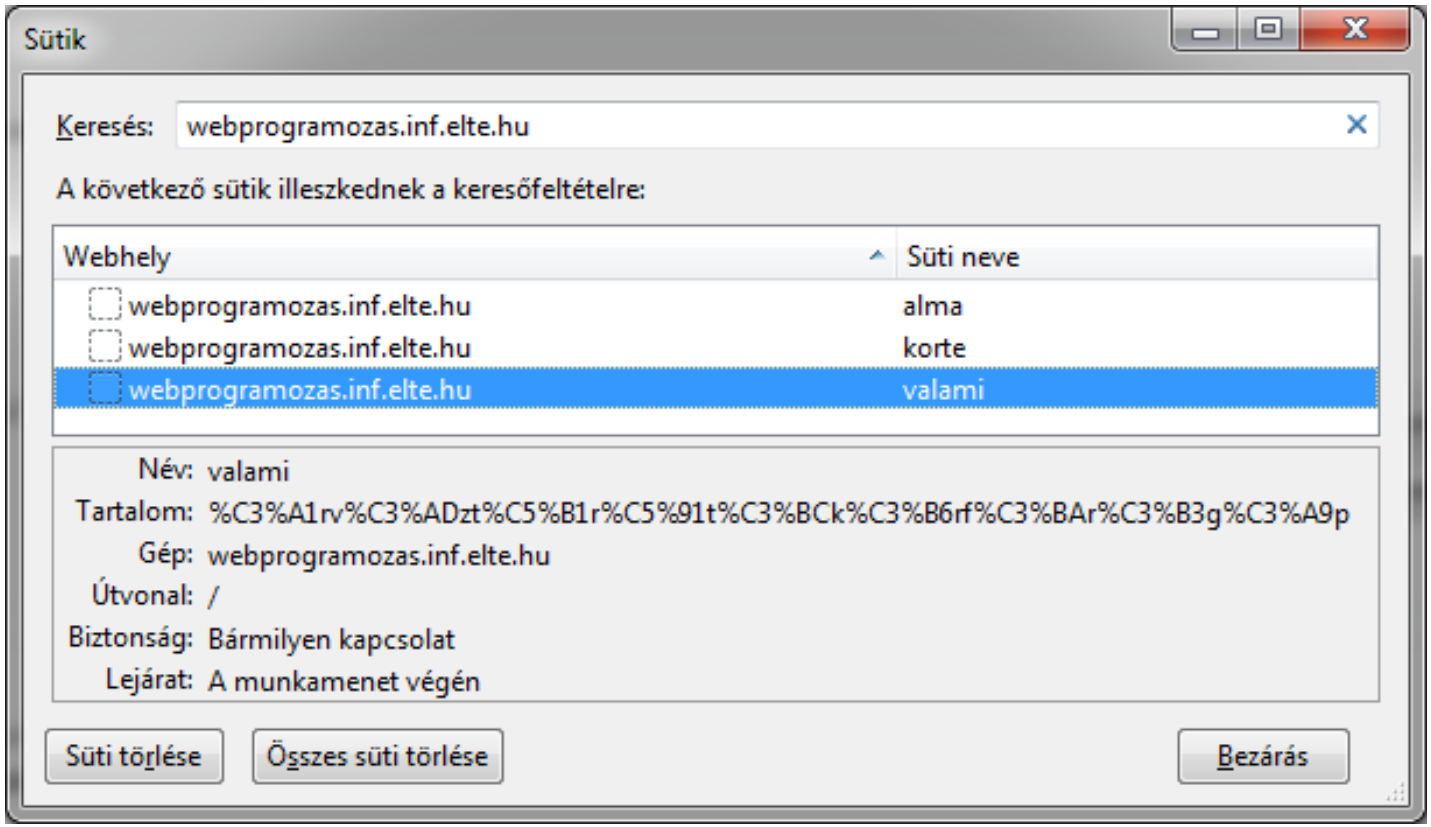
```
function sutiBeallit(nev, ertek, lejarat) {
  document.cookie = nev + '=' + encodeURIComponent(ertek) +
    (lejarat ? '; expires=' + lejarat.toGMTString() : '');
}
```

- Süti törlése

```
function sutiTorol(nev) {
        sutiBeallit(nev, '', new Date());
}
```

Böngésző


```
sutiBeallit('alma', 'piros');
sutiBeallit('korte', 'sárga');
sutiBeallit('valami', 'árvíztűrőtükörfúrógép');
```



Segédfüggvények

- Süti lekérdezése = document.cookie feldolgozása

```
console.log(document.cookie);
//"alma=piros; korte=s%C3%A1rga;
valami=%C3%A1rv%C3%ADzt%C5%B1r%C5%91t%C3%BCk%C3%B6rf%C3%BA
r%C3%B3g%C3%A9p"
```

- Többféle feldolgozás
  - felbontás ; és = szerint
  - szövegkeresés
  - reguláris kifejezés

Segédfüggvények

- Lekérdezés felbontással

```
"alma=piros; korte=s%C3%A1rga; valami=valami"
```

```
function sutiLeker(nev) {
  var c = document.cookie;
  if (!c) return undefined;
  var sutik = c.split(';');
  for (var i = 0; i<sutik.length; i++) {
    var nevertek = sutik[i].split('=');
    nevertek[0] = nevertek[0].replace(/^\s*/, '');
    if (nevertek[0] == nev) {
      return decodeURIComponent(nevertek[1]);
    }
  }
  return undefined;
}
```

Segédfüggvények

- Lekérdezés szövegkereséssel

```
"alma=piros; korte=s%C3%A1rga; valami=valami"
```

```
function sutiLeker(nev) {
  var c = document.cookie;
  var nevpoz = c.search(new RegExp('(^|;)\\s*'+nev));
  if (nevpoz>=0) {
    var ertekpoz = c.indexOf('=', nevpoz);
    ertekpoz += 1;
    var vegepoz = c.indexOf(';', ertekpoz);
    if (vegepoz == -1) {
      vegepoz = c.length;
    }
    var ertek = c.substring(ertekpoz, vegepoz);
    return decodeURIComponent(ertek);
  }
        return undefined;
}
```

Segédfüggvények

- Lekérdezés reguláris kifejezéssel

```
"alma=piros; korte=s%C3%A1rga; valami=valami"
```

```
function sutiLeker(nev) {
  nev = decodeURIComponent(nev);
  return decodeURIComponent(
    document.cookie.replace(
      new RegExp('(^|.*;)\\s*' + nev + '\\s*=\\s*([^;]*).*$'),
      "$2"
    )
  );
}
```

Hivatkozások

- https://developer.mozilla.org/en-US/docs/DOM/document.cookie
- http://www.nczonline.net/blog/2009/05/05/http-cookies-explained/

HTML 5 API-k

Néhány HTML 5 JavaScript API

- Audio, video
- Canvas
- Locale storage
- Offline Apps
- Web worker
- Drag and drop
- Contenteditable
- Web sockets, server events
- Geolocation

Audio

- Hangállományok lejátszása
- HTML <audio> tag
  - src
  - controls
  - autoplay
  - stb.

```
<audio src="horn3.wav" id="audio1" controls></audio>
```

Audio

- JavaScript – meglévő elem manipulálása
  - HTMLAudioElement
  - Elérve az elemet rengeteg tulajdonsága van
  - Főbb metódusok: play(), pause()

```
<audio src="horn3.wav" id="audio1"></audio>
```

```
$('audio1').play();
```

- JavaScript – új elem létrehozása

```
var a = document.createElement('audio');
a.src = 'horn3.wav';
//...
a.play();
```

Video

- Videóállományok lejátszása
- HTML <video> tag
- Sokféle formátum
- Sokféle tulajdonság
- JavaScript
  - HTMLVideoElement
- Audiohoz teljesen hasonló programozás

Canvas

- Böngészők grafikus lehetőségei
  - CSS
  - Képek
  - SVG (vektorgrafika)
  - Canvas (raszteres grafika)
- HTML5: <canvas> elem
  - Raszteres grafika natív támogatása
  - + JavaScript API

Canvas

- HTML <canvas> elem
  - szélesség
  - magasság
  - záró elem </canvas>
  - ha nincs támogatás

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

```
<canvas id="canvas" width="200" height="200">
  Ez akkor jelenik meg, ha nincs canvas támogatás
  szöveg, kép, stb.
</canvas>
```

Canvas

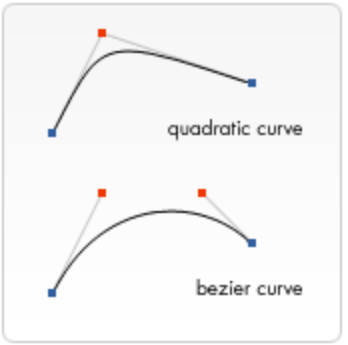
- JavaScript: rajzolói környezet kiválasztása
  - 2d
  - 3d (webgl, opengl)

```
var canvas = document.getElementById('canvas');
var ctx = canvas.getContext('2d');
```

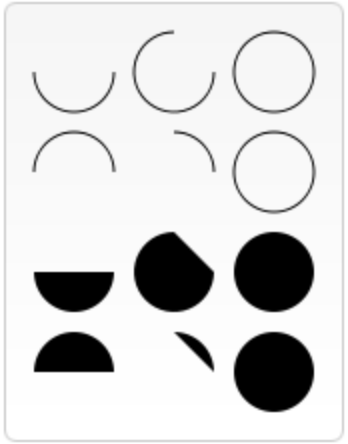
- Támogatás ellenőrzése

```
var canvas = document.getElementById('canvas');
if (canvas.getContext){
  var ctx = canvas.getContext('2d');
  //rajzolás
} else {
  //nincs canvas támogatás
}
```

Canvas – alakzatok



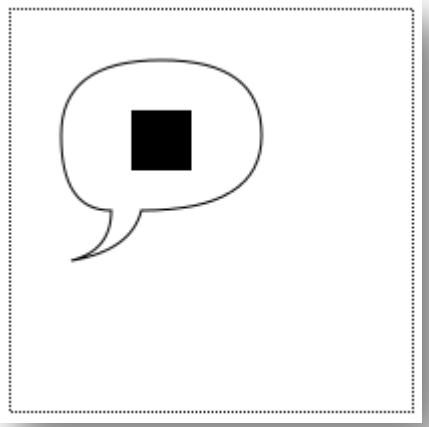
- Téglalap
  - **fillRect**(x,y,width,height)
    strokeRect(x,y,width,height)
    clearRect(x,y,width,height)
- Útvonalak
  - **beginPath**()
    closePath()
    stroke()
    fill()
- Vonalak
  - **moveTo**(x, y)
    lineTo(x, y)



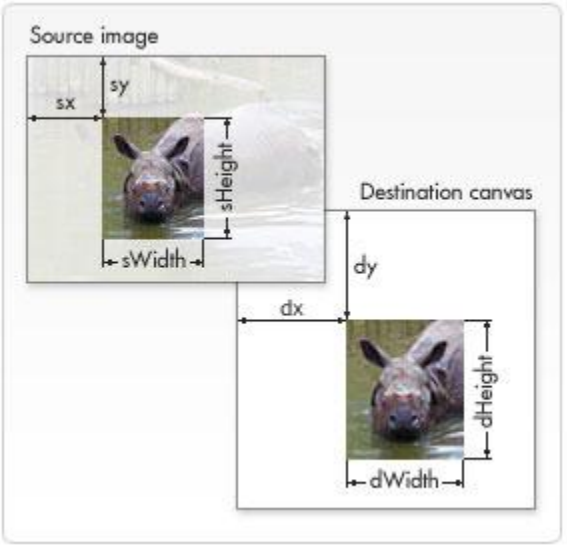
- Ívek
  - **arc**(x, y, radius, startAngle, endAngle, anticlockwise)
- Bezier-görbék
  - **quadraticCurveTo**(cp1x, cp1y, x, y)
    bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y)

Példa


```
//Téglalap rajzolása
ctx.fillRect(60, 50, 30, 30);
//Szövegbuborék rajzolása
ctx.beginPath();
ctx.moveTo(75,25);
ctx.quadraticCurveTo(25,25,25,62.5);
ctx.quadraticCurveTo(25,100,50,100);
ctx.quadraticCurveTo(50,120,30,125);
ctx.quadraticCurveTo(60,120,65,100);
ctx.quadraticCurveTo(125,100,125,62.5);
ctx.quadraticCurveTo(125,25,75,25);
ctx.stroke();
```



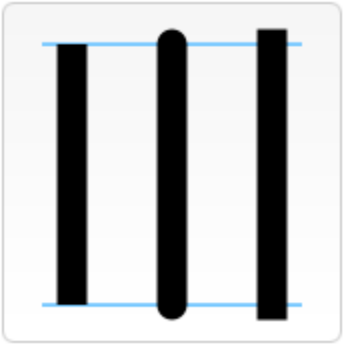
Canvas - Képek



- Forrás
  - meglévő kép felhasználása: <img>
  - memóriabeli kép: var img=document.createElement('img');
  - másik canvas
- Műveletek
  - **drawImage**(image, x, y)
  - **drawImage**(image, x, y, width, height)
  - **drawImage**(image, sx, sy, sWidth, sHeight, dx, dy, dWidth, dHeight)

Canvas – stílusok és színek

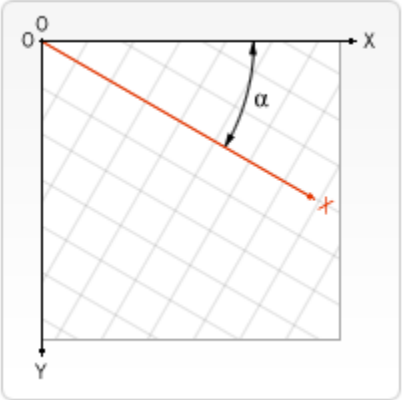
- **fillStyle** = color
  strokeStyle = color
  - ```
    ctx.fillStyle = "orange";
    ctx.fillStyle = "#FFA500";
    ctx.fillStyle = "rgb(255,165,0)";
    ctx.fillStyle = "rgba(255,165,0,1)";
    ```
- **globalAlpha** = transparency value
- Vonalstílusok
- Gradiensek
- Mintázatok
- Árnyékolás

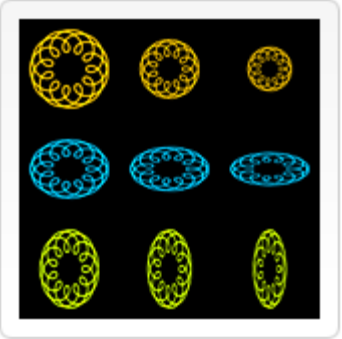



    ```

Canvas - Transzformációk

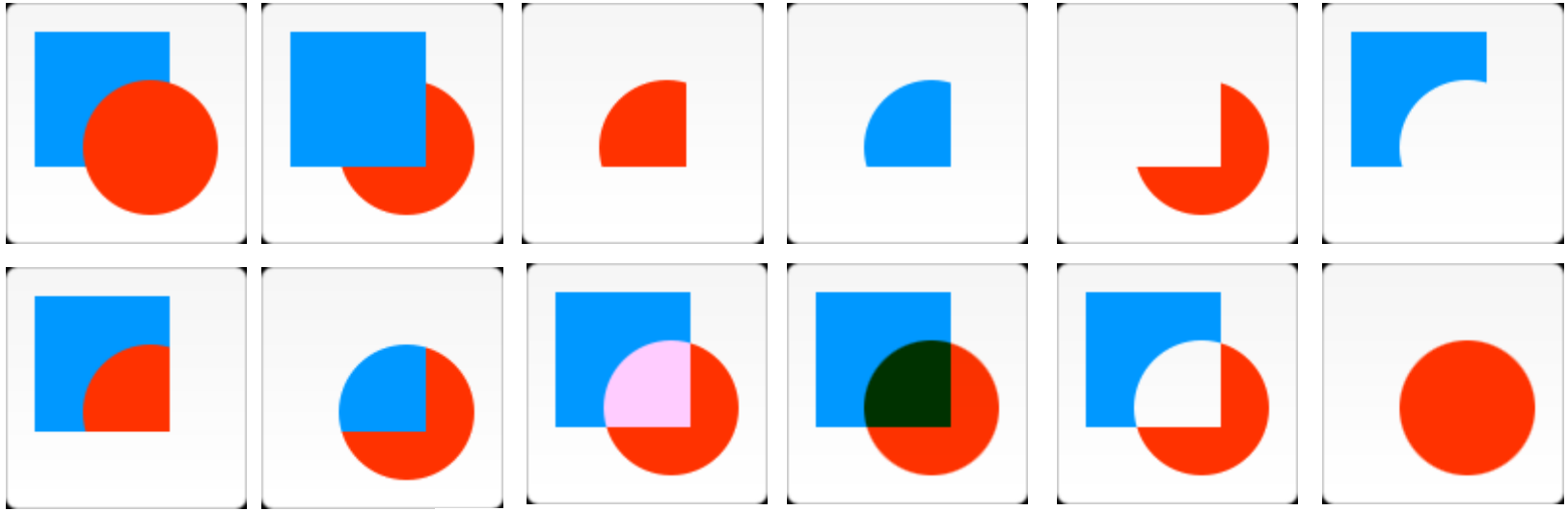
- Állapotkezelés
  - **save()**
    restore()
- Transzformációk
  - **translate(x, y)**
  - **rotate(angle)**
  - **scale(x, y)**
  - **transform(m11, m12, m21, m22, dx, dy)**



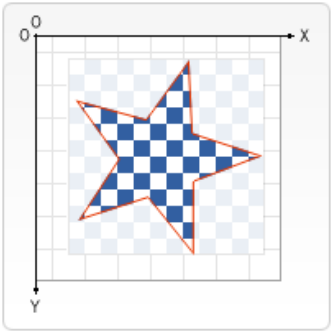


Canvas – Egyesítés és vágás

- Egyesítés
  - **globalCompositeOperation** = type



- Vágás
  - útvonal
  - **clip**()



Canvas – animációk

- Lépések
  - canvas törlése
  - állapotmentés
  - rajzolás
  - állapot visszaállítása
- setInterval(animateShape,500);
  setTimeout(animateShape,500);
- requestAnimationFrame(animateShape)

Animáció – HTML


```
<!doctype html>
<html>
  <head>
    <title></title>
    <script type="text/javascript" src="sprite.js"></script>
  </head>
  <body>
    <canvas id="myCanvas" width="512" height="512"></canvas>
    <!-- <img src="sprite.png"> -->
    <div id="fps"></div>
  </body>
</html>
```

```
window.addEventListener('load', init, false);
function init() {
    canvas = document.getElementById('myCanvas');
    ctx = canvas.getContext("2d");

    img = new Image();
    img.src = 'sprite.png';
    img.addEventListener('load', loaded, false);
}

var canvas;
var ctx;
var img;

function loaded() {
    draw();
}
function draw() {
    ctx.fillStyle = '#000000';
    ctx.fillRect(0, 0, canvas.width, canvas.height);
    ctx.drawImage(img, 0, 0);

    ctx.drawImage(img,
        0, 0, 127, 127,
        canvas.width / 2 - 32, canvas.height / 2 - 32, 64, 64);
}
```

Játékciklus


```
window.requestAnimFrame = (function() {
  return window.requestAnimationFrame ||
    window.webkitRequestAnimationFrame ||
    window.mozRequestAnimationFrame ||
    window.oRequestAnimationFrame ||
    window.msRequestAnimationFrame ||
    function(callback) {
      window.setTimeout(callback, 1000 / 60);
  };
})();

function loaded() {
  mainloop();
}

var mainloop = function() {
  requestAnimFrame(mainloop);
  update();
  draw();
};
```

Játékciklus

```
var fr = 0;
var startTime = new Date();

var frame = 0;
var numberOfFrames = 16;

function update() {
  //FPS
  fr++;
  document.getElementById('fps').innerHTML = Math.floor(fr * 1000 / (new
Date() - startTime));
  //Frame
  frame = (frame + 1) % numberOfFrames;
}

function draw() {
  ctx.fillStyle = '#000000';
  ctx.fillRect(0, 0, canvas.width, canvas.height);

  ctx.drawImage(img,
    (frame % 4) * 128, Math.floor(frame / 4) * 128, 127, 127,
    canvas.width / 2 - 64, canvas.height / 2 - 64, 128, 128);
}
```

Idő alapú animáció


```
var frame = 0;
var numberOfFrames = 16;
var lastTime = Date.now();
var cumDelta = 0;
var msPerFrame = 30;

function update() {
    //FPS
    //...
    //frame
    var most = Date.now();
    var delta = most - lastTime;
    if (cumDelta + delta > msPerFrame) {
        cumDelta = 0;
        frame = (frame + 1) % numberOfFrames;
    } else {
        cumDelta += delta;
    }
    lastTime = most;
}
```

Mozgás


```
var x = 0;
var vx = 150;
var irany = 1;

function update() {
  //...
  if (x > canvas.width - 64) {
    irany = -1;
  } else if (x < 0) {
    irany = 1;
  }
  x += vx * irany * delta / 1000;
}
function draw() {
  //...
  ctx.save();
  ctx.translate(x + 32, 200 + 32);
  ctx.scale(0.5 * irany, 0.5);
  ctx.drawImage(img,
    (frame % 4) * 128, Math.floor(frame / 4) * 128, 127, 127,
    -32, -32, 128, 128);
  ctx.restore();
}
```

Felhasználói objektumok

Felhasználói objektumok, öröklés

Objektumliterál

- Egyedi objektumok megadása

```
var ubul = {
  nev: 'Ubul',
  kor: 13,
  bemutatkozik: function () {
    console.log('A nevem: ' + this.nev);
  },
  alszik: function () {
    console.log('Zzzzzz...');
  }
};

console.log(ubul.nev);
ubul.bemutatkozik();
ubul.alszik();
```

Objektumgenerátor függvény

- Ha több ugyanolyan objektumra van szükség

```
function ember(nev, kor) {
  return {
    nev: nev,
    kor: kor,
    bemutatkozik: function () {
      console.log('A nevem: ' + this.nev);
    },
    alszik: function () {
      console.log('Zzzzzz...');
    }
  };
}

var ubul = ember('Ubul', 13);
var dome = ember('Döme', 8);
ubul.bemutatkozik();
dome.bemutatkozik();
```

Objektumkonstruktor függvény

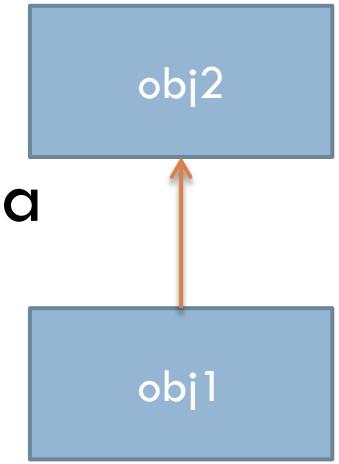
- Objektumorientált-jellegű szintaxis

```
function Ember(nev, kor) {
  this.nev = nev;
  this.kor = kor;
  this.bemutatkozik = function () {
    console.log('A nevem: ' + this.nev);
  };
  this.alszik = function () {
    console.log('Zzzzzz...');
  };
}

var ubul = new Ember('Ubul', 13);
var dome = new Ember('Döme', 8);
ubul.bemutatkozik();
dome.bemutatkozik();
```

Prototípus

- JavaScriptben minden objektumnak van egy rejtett hivatkozása, amin keresztül egy objektum tartozik hozzá
- Prototípus objektum
- Ebben lévő tulajdonságok és függvények elérhetők
- Prototípusnak is lehet prototípusa
- →prototípus láncolat
- Konstruktorfüggvény prototype tulajdonsága



Objektumkonstruktor függvény

- Hatékonyabb metódusmegadás

```
function Ember(nev, kor) {
  this.nev = nev;
  this.kor = kor;
}
Ember.prototype.bemutatkozik = function () {
  console.log('A nevem: ' + this.nev);
};
Ember.prototype.alszik = function () {
  console.log('Zzzzzz...');
};

var ubul = new Ember('Ubul', 13);
var dome = new Ember('Döme', 8);
ubul.bemutatkozik();
dome.bemutatkozik();
```

Öröklés

- Konstruktorfüggvénnyel

```
function Gyerek(nev, kor, jel) {
  this.base = Ember;
  this.base(nev, kor);
  this.jel = jel;
}
Gyerek.prototype = new Ember();
Gyerek.prototype.miAJeled = function () {
  console.log('A jelem: ' + this.jel);
}

var ubul = new Ember('Ubul', 13);
ubul.bemutatkozik();
var bruno = new Gyerek('Brúnó', 3, 'labda');
bruno.bemutatkozik();
bruno.miAJeled();
```

Öröklés

- ECMAScript 5: Object.create() metódussal

```
var ubul = {
  nev: 'Ubul',
  kor: 13,
  bemutatkozik: function () {
    console.log('A nevem: ' + this.nev);
  },
  alszik: function () {
    console.log('Zzzzzz...');
  }
};
```

```
var bruno = Object.create(ubul);
bruno.jel = 'labda';
bruno.miAJeled = function () {
  console.log('A jelem: ' + this.jel);
}

ubul.bemutatkozik();
ubul.alszik();
bruno.bemutatkozik();
bruno.miAJeled();
```

Hivatkozások

- https://developer.mozilla.org/en-US/docs/JavaScript/Guide/Working_with_Objects
- https://developer.mozilla.org/en-US/docs/JavaScript/Guide/Details_of_the_Object_Model
- https://developer.mozilla.org/en-US/docs/JavaScript/Guide/Inheritance_Revisited

Kivételkezelés

Hibák

- Error
  - EvalError
  - RangeError
  - ReferenceError
  - SyntaxError
  - TypeError
  - URIError
- Tulajdonságok
  - name
  - message

Hibakezelés

- try-catch-finally
  - try: védendő kód
  - catch: hibakezelő kód
  - finally: a végén lefutó kód (nem kötelező)

```
try {
  alma.kukacos = true;
}
catch (e) {
  console.log(e.name);    //ReferenceError
  console.log(e.message); //alma is not defined
}
finally {  //Elhagyható
  console.log('Végem van...');
}
```

Hiba dobása

- Beépített hiba dobása

```
if (typeof a !== 'number') {
  throw new Error('Nem szam a parameter!');
}
```

Hiba dobása

- Saját hiba dobása

```
if (oszto == 0) {
  throw {
    name: 'DivisionByZeroError',
    message: 'Az oszto nulla!'
  };
}
```

- vagy

```
function DivisionByZeroError(message) {
  this.name = "DivisionByZeroError";
  this.message = message;
}
//...
if (oszto == 0) {
  throw new DivisionByZeroError('Az oszto nulla!');
}
```

Kódszervezés

Kódszervezés

- Sok függvény keletkezik a megoldás során
- Csoportosítás
  - megjegyzésekkel
  - logikai
  - fizikai

Megjegyzések


```
//Segédfüggvények
function $(id) {
  return document.getElementById(id);
}
//Feldolgozó függvények
function nevbolUdvozles(nev) {
  return 'Hello ' + nev + '!';
}
//Eseménykezelő függvények
function hello() {
  var nev = $('nev').value;
  var udvozles = nevbolUdvozles(nev);
  $('kimenet').innerHTML = udvozles;
}
function init() {
  //Eseménykezelők regisztrálása
  $('gomb').addEventListener('click', hello, false);
}
window.addEventListener('load', init, false);
```

Logikai csoportosítás – névterek

- „Névtér”

```
hello.nevbolUdvozles(nev);
```

- Objektumok

```
hello = {
  nevbolUdvozles: function(nev) { /*...*/ },
  hello: function() { /*...*/ },
  init: function() { /*...*/ }
};
```

Logikai csoportosítás – névterek


```
//Segédfüggvények
var function $(id) {
        return documengetElementById(id);
}
hello = {};
//Feldolgozó függvények
hello.nevbolUdvozles = function(nev) {
        return 'Hello ' + nev + '!';
}
//Eseménykezelő függvények
hello.hello = function() {
        var nev = $('nev').value;
        var udvozles = hello.nevbolUdvozles(nev);
        $('kimenet').innerHTML = udvozles;
}
//Belépési pont
hello.init = function() {
        //Eseménykezelők regisztrálása
        $('gomb').addEventListener('click', hello.hello, false);
}
window.addEventListener('load', hello.init, false);
```

Fizikai csoportosítás

- Több névtér esetén
- Logikai csoportonként → fájl
- → hello.js

JSON

JSON

- JavaScript Object Notation
- A JavaScript literálformáira épülő adatleírási formátum
- Központ eleme
  - objektum: {}
  - tömb: []
- Elterjedt
- JSON.stringify()
- JSON.parse()

JSON példa

- Szótáras feladat

```
[
    { "angol": "apple", "magyar": "alma"   },
    { "angol": "pear",  "magyar": "korte"  },
    { "angol": "plum",  "magyar": "szilva" },
    { "angol": "peach", "magyar": "barack" }
]
```

Esettanulmány

Grafilogika

- http://webprogramozas.inf.elte.hu/gyak/js_grafilogika.html
- Feladat elemzése
  - HTML szükséglet
  - Adatszükséglet
- Pálya leírása → mátrix (abra.js)
- Pálya működtetése → mátrix
- kiíró függvények
- feldolgozó függvények
- eseménykezelő függvények