WEBFEJLESZTÉS 2. – NYELVI ELEMEK, IDŐZÍTŐK, STÍLUSOK, ANIMÁCIÓK

Horváth Győző

Egyetemi adjunktus

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

JavaScript beadandó

- Labirintus
- http://webprogramozas.inf.elte.hu/gyak/js_labirintus.html

Ismétlés

Ismétlés

- JavaScript nyelvi elemei
- JavaScript és HTML kapcsolata
- Eseménykezelés részletei
- JavaScript beépített objektumok
- DOM
- HTML DOM: űrlapok, képek, táblázatok
- BOM

Névtelen függvények

Névtelen függvény


```
function init() {
  //...
}
window.addEventListener('load', init, false);

//HELYETTE

window.addEventListener('load', function () {
  //...
}, false);
```

Függvényliterál


```
//Függvényliterál
function (par1, par2) {
  //...
}

//Paraméteres eseménykezelők
$('gomb1').addEventListener('click', function (e) {
  leptet(e, -1);
}, false);

//Függvény létrehozása
var fv = function (a, b) {
  return a + b;
}
console.log( fv(18, 24) );
```

Reguláris kifejezések

Reguláris kifejezés

- Egy olyan (szövegként megadott) minta, amely szövegekre illeszthető
- Speciális karakterek szerepelnek benne
- Például:

```
console.log(/\d+/.test('alma'));    //false
console.log(/\d+/.test(''));        //false
console.log(/\d+/.test('123'));     //true
console.log(/\d+/.test('alma123')); //true
```

Reguláris kifejezés

- Literálforma: /minta/flags
  - minta: szöveg, normál szöveg + speciális jelölők
  - flag: viselkedést szabályozó jelölők
- Metódusok
  - test() → logika érték
  - exec() → tömb
- String függvények használják intenzíven
  - search(regexp)
  - replace(regexp, mire)

Minta

- Normál szöveg:

```
console.log(/alma/.test('piros alma kukacos')); //true
```

- ^ elejére illeszkedik
- $ végére illeszkedik
- * legalább 0-szor ismétlődik
- + legalább 1-szer ismétlődik
- ? 0 vagy 1-szer ismétlődik
- {n,m} legalább n-szer, legfeljebb m-szer ismétlődik

Minta

- . bármilyen karakter
- (minta) csoportosítás
- \d számok: [0-9]
- \D nem számok
- \s fehér szóköz
- \w alfanumerikus karakter: [A-Za-z0-9_]

```
console.log(/alma$/.test('piros alma kukacos')); //false
console.log(/alma$/.test('piros alma'));         //true
console.log(/\w+$/.test('piros alma'));          //true
```

Flag-ek

- g: globális egyezés, ne álljon meg az elsőnél
- i: nem érzékeny a kis és nagybetűkre
- m: multiline – sorvégjeleken átívelő egyezés

```
console.log(/alma/.test('Piros Alma'));          //false
console.log(/alma/i.test('Piros Alma'));         //true
console.log("piros alma kukacos".replace(/os/, 'itott'));
//"piritott alma kukacos"
console.log("piros alma kukacos".replace(/os/g, 'itott'));
//"piritott alma kukacitott"
```

Metódusok

- regexp.test(szöveg)
  - megvizsgálja illeszkedik a minta a paraméterként megadott szövegre
- regexp.exec(szöveg)
  - az illeszkedés információit egy tömbben adja vissza

```
console.log(/os/.exec('piros alma kukacos'));
```

```
>>> console.log(/os/.exec('piros alma kukacos'));
[ "os" ]
  0                    "os"
  index                3
  input                "piros alma kukacos"
```

Szövegműveletek

- szöveg.search(regexp)
- szöveg.replace(regexp, mire)

```
var re = /(\w+)\s(\w+)/i;
var sz = "Horvath Gyozo";
console.log(sz.replace(re, "$2, $1"));
//Gyozo, Horvath
```

Hivatkozások

- https://developer.mozilla.org/en-US/docs/JavaScript/Reference/Global_Objects/RegExp
- https://developer.mozilla.org/en-US/docs/JavaScript/Guide/Regular_Expressions

Időzítő

setTimeout, setInterval

Időzítő

- Egyszeri
  - egy függvény valahány ms múlva végrehajtódik
- Ismételt
  - egy függvény valahány ms-onként végrehajtódik
- Aszinkron
  - végrehajtás tovább megy
  - felület zavartalanul működik

Időzítő – egyszeri

- setTimeout, clearTimeout általában

```
var idozito = setTimeout(fuggveny, ms);
clearTimeout(idozito);
```

- setTimeout példa

```
function csorog() {
  console.log('Brrrrrrr');
}
setTimeout(csorog, 2000);
```

- clearTimeout példa

```
function csorog() {
  console.log('Brrrrrrr');
}
var idozito = setTimeout(csorog, 2000);
clearTimeout(idozito);
```

Időzítő – ismétlődő

- setInterval, clearInterval általában

```
var idozito = setInterval(fuggveny, ms);
clearTimeout(idozito);
```

- setInterval példa

```
function csorog() {
  console.log('Brrrrrrr');
}
setInterval(csorog, 2000);
```

- clearInterval példa

```
function csorog() {
  console.log('Brrrrrrr');
}
var idozito = setInterval(csorog, 2000);
clearInterval(idozito);
```

Stílusok

style, class, stylemaps

Stílusmanipulációk

- CSS szabályok elengedhetetlen ismerete szükséges
- Három szint
  - style attribútum
  - class attribútum
  - stíluslapok

Style attribútum

- HTML: style attribútum

```
<elem style="css szabály"></elem>
```

- JavaScript: az elemet style tulajdonsága

```
elem.style
```

- Az elem.style egy objektum (CSSProperties), aminek egyes tulajdonságain keresztül kérdezhetjük le vagy állíthatjuk be a stílustulajdonságokat.
- Analóg módszer szerinti megfeleltetés

Style objektum tulajdonságai

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

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

Style objektum

- Csak azok a stílustulajdonságok kérdezhetők le, amelyek
  - a style attribútumon keresztül voltak megadva;
  - JavaScriptből határoztuk meg az értéküket.
- Nem kérdezhető le akármilyen tulajdonság így.
- → számított stílus

Számított stílus

- Kiolvashatók azonban azok a tulajdonságok, amelyeket a böngésző tart nyilván az elemekről
- Szabványos
  - window.getComputedStyle(elem)
- Internet Explorer (régebbiek)
  - elem.currentStyle()
- A rövidítések (pl. border, background, stb.) nem érhető el, csak az elemi tulajdonságok.

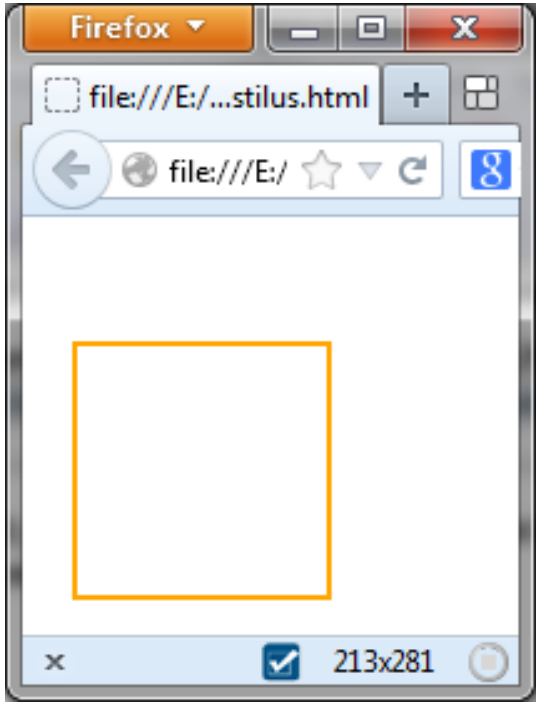
Példa


```
<style type="text/css">
.doboz {
  width: 100px;
  height: 100px;
  border: 2px solid orange;
  position: absolute;
}
</style>
<div id="doboz" class="doboz"
style="left: 20px"></div>
```

```
var d = document.getElementById('doboz');
console.log(d);
d.style.top = '50px';

console.log(d.style);          //CSS2Properties
console.log(d.style.top);      //"50px"
console.log(d.style.left);     //"20px"
console.log(d.style.width);    //""
console.log(d.style.height);   //""

var s = window.getComputedStyle(d);
console.log(s.top);            //"50px"
console.log(s.left);           //"20px"
console.log(s.width);          //"100px"
console.log(s.height);         //"100px"
console.log(s.border);         //""
console.log(s.borderBottomWidth);    //"2px"
console.log(s.position);       //"absolute"
```



Class attribútum

- HTML: class attribútum

```
<style type="text/css">
  .osztaly { ... }
  .osztaly1 { ... }
  .osztaly2 { ... }
  .osztaly3 { ... }
</style>
<elem class="osztaly"></elem>
<elem class="osztaly1 osztaly2 osztaly3"></elem>
```

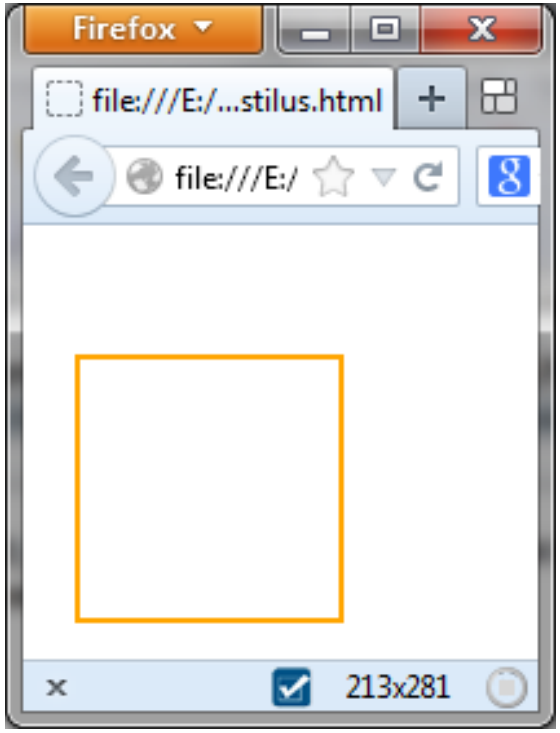
- JavaScript: className tulajdonság
  - Szöveges érték

```
console.log(elem.className); //lekérdezés
elem.className = 'osztaly';  //beállítás
```

Class példa


```
<style type="text/css">
.doboz {
        width: 100px;
        height: 100px;
        border: 2px solid orange;
        position: absolute;
        top: 50px;
        left: 20px;
}
</style>

<div id="doboz"></div>
```




```
var d = document.getElementById('doboz');
d.className = 'doboz';
console.log(d.className);  //doboz
```

Stílusosztály-kezelés

- Mit tegyünk, ha több osztály van megadva, és csak egy osztályt szeretnénk hozzáadni vagy elvenni?
  - Szövegfeldolgozás
  - Tömbfeldolgozás, indexOf, splice
- Segédfüggvények
  - Van-e adott stílusosztály beállítva?
  - Hozzáad stílusosztályt
  - Elvesz stílusosztályt

```
<div class="piros barna sarga">
```

Segédfüggvények


```
function vaneOsztaly(elem, o) {
  return elem.className.search(new RegExp('(^|\\s+)' + o + '(\\s+|$)')) > -1;
}
function hozzaadOsztaly(elem, o) {
  if (!vaneOsztaly(elem, o)) {
    elem.className += (elem.className ? ' ' : '') + o;
  }
}
function torolOsztaly(elem, o) {
  elem.className = elem.className
    //osztálynév törlése
    .replace(new RegExp('(^|\\s+)' + o + '(\\s+|$)'), ' ')
    //kétoldali felesleges szóköz eltávolítása
    .replace(/^\s*|\s*$/g, '');
}
```

```
<div class="piros barna sarga">
```

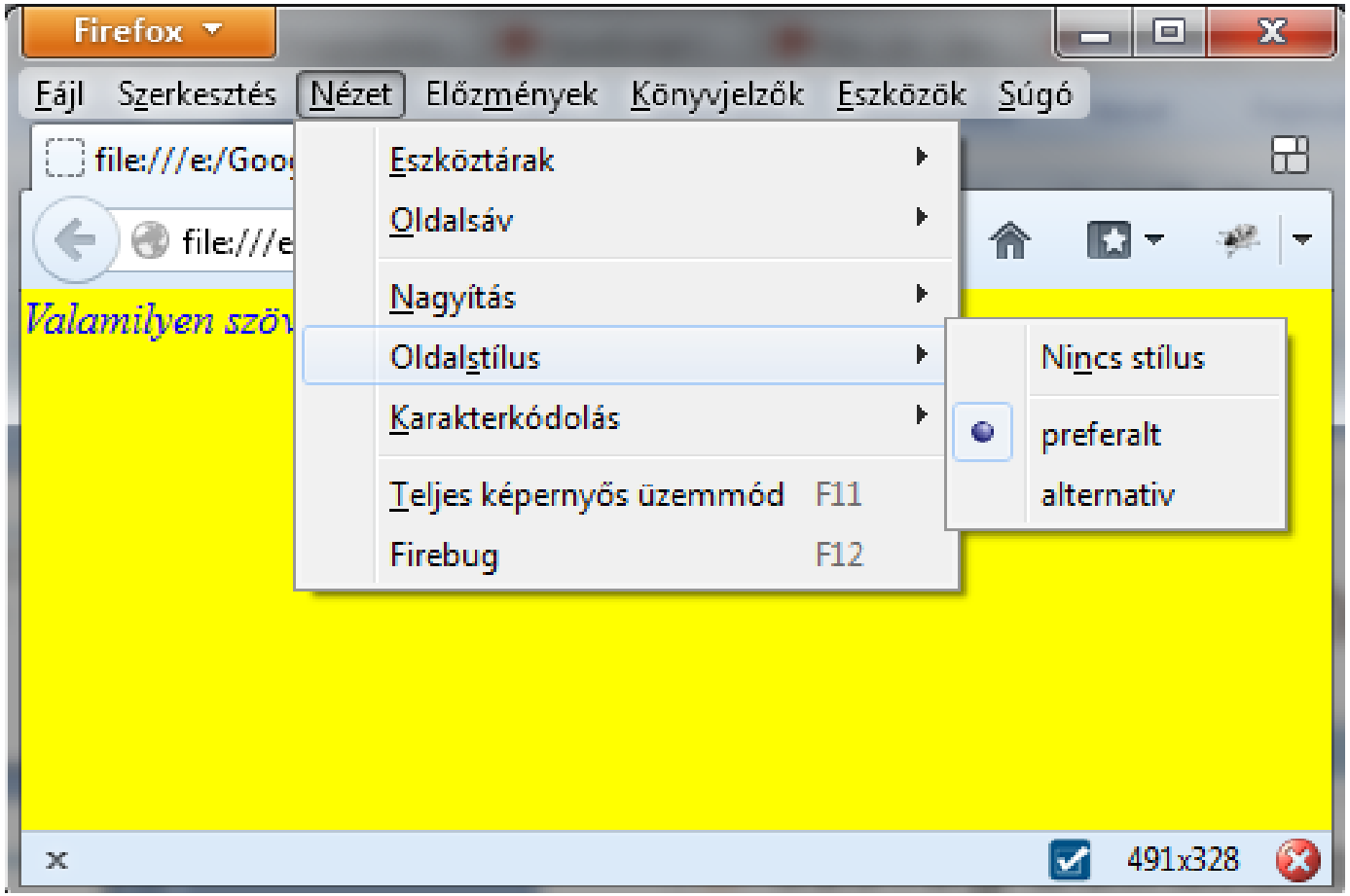
Stíluslapok

- HTML: link vagy style elem
- Helye szerint
  - Belső stíluslap (style elem)
  - Külső stíluslap (link elem)
- Típusa szerint
  - permanens (mindig aktív, nem kikapcsolható)
  - preferált (betöltéskor aktív, de ki-bekapcsolható)
  - alternatív stíluslap (betöltéskor nem aktív, ki-bekapcsolható)

Stíluslapok – példa


```
<html>
  <head>
    <meta charset="utf-8">
    <title></title>
    <link rel="stylesheet" type="text/css" href="permanens.css">
    <link rel="stylesheet" type="text/css" href="preferalt.css"
title="preferalt">
    <link rel="alternate stylesheet" type="text/css"
href="alternativ.css" title="alternativ">
    <style type="text/css">
      p {
        font-style: italic;
      }
    </style>
  </head>
  <body>
    <p>Valamilyen szöveg.</p>
  </body>
</html>
```

Stíluslapok – példa



Stíluslapok

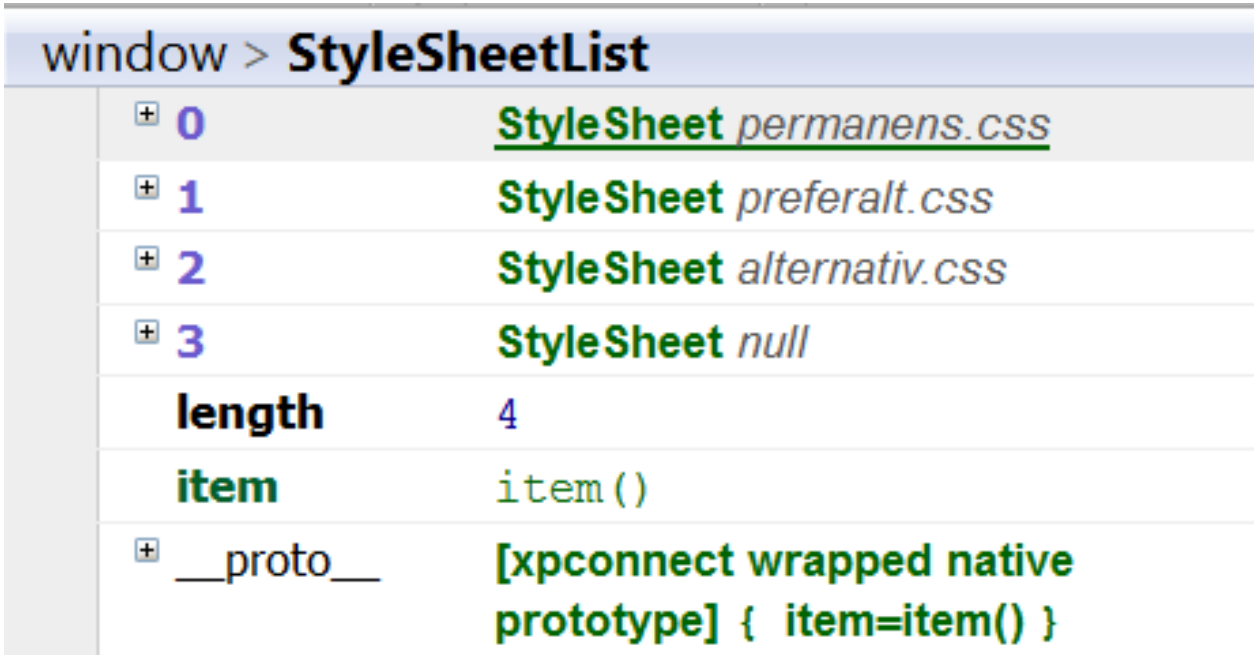
- JavaScript: document.styleSheets
- Tömb – StyleSheet-ekből áll
- StyleSheet
  - type
  - disabled (stíluslap dinamikus ki-bekapcsolása)
  - href
  - title
- Szabványos (DOM Level 2)

Stíluslapok

- A stíluslapon belüli szabályok is elérhetők
  - szabványos: cssRules tömb
  - Internet Explorer: rules tömb
- Minden szabálynál
  - cssText: a szabály szöveges formája
  - selectorText (nem szabványos)
  - style: az adott szabály stíluslistája (mint az inner style)
- Ezekkel
  - új szabály vehető fel
  - régi törölhető
  - meglévő módosítható

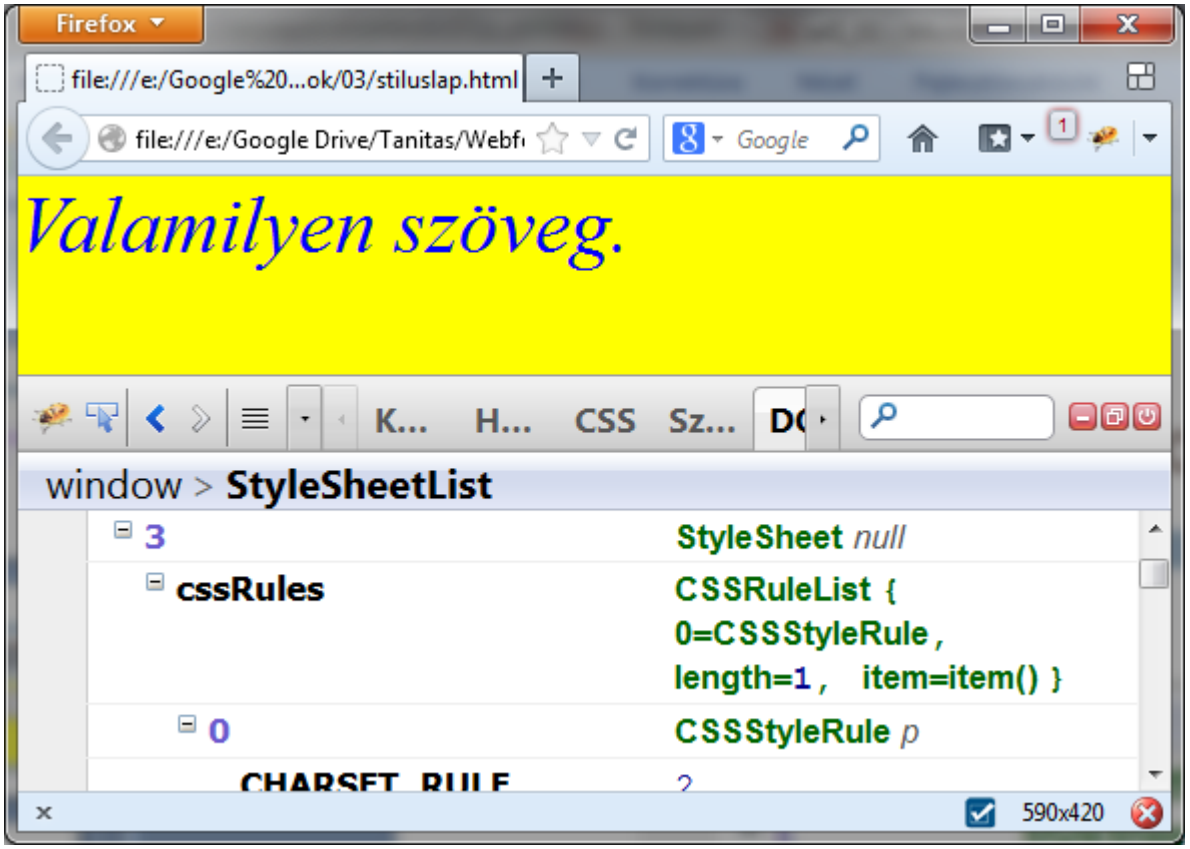
Stíluslapok – példa

- document.styleSheets



Stíluslapok – példa


```
document.styleSheets[3].cssRules[0].style.fontSize = '40px';
```



Animációk

JavaScript, CSS3

Animációk

- Elem adott stílusának kezdő és végállapota közötti adott idő alatt lejátszódó folyamatos(nak tűnő) átmenet.
- Nem rögtön a végállapotot állítjuk be, hanem az értékintervallumot diszkrét lépésekre bontjuk, és ezen értékeket adjuk az elem stílustulajdonságának.
- Hogyan?
  - Ciklus: nagyon gyors, felfüggeszti a felületet
  - Időzítő: aszinkron módon, bizonyos időközönként

Animáció példa

- div szélességét állítsuk 500px-re
- Azonnali megoldás

```
function binaris() {
  var d = document.getElementById('doboz');
  d.style.width = '500px';
}
```

- Ciklus

```
function ciklus() {
  var d = document.getElementById('doboz');
  var w = parseInt(window.getComputedStyle(d).width, 10);
  for (var i = w; i<=500; i++) {
    d.style.width = i + 'px';
  }
}
```

Animáció JavaScripttel

- Bizonyos időközönként változtatni a stílustulajdonságot
- → időzítő használata
- Felület használható marad
- További szkriptek futnak
- Történetileg az első megoldás
- Paraméterei
  - időtartam
  - (kezdőérték)→végértek
  - stílustulajdonság
  - hány lépésben (→milyen lépésközzel)

Animáció – példa időzítővel


```
function idozito() {
  var d = document.getElementById('doboz');
  var w = parseInt(window.getComputedStyle(d).width, 10);
  if (w < 500) {
    d.style.width = (w + 10) + 'px';
    setTimeout(idozito, 50);
  }
}
```

Animáció JavaScripttel

- Hátrányai
  - Sok elem mozgatása lassú
  - Jól kell megválasztani a lépésszámot
  - Nehezebb optimalizálni

CSS3 lehetőségek

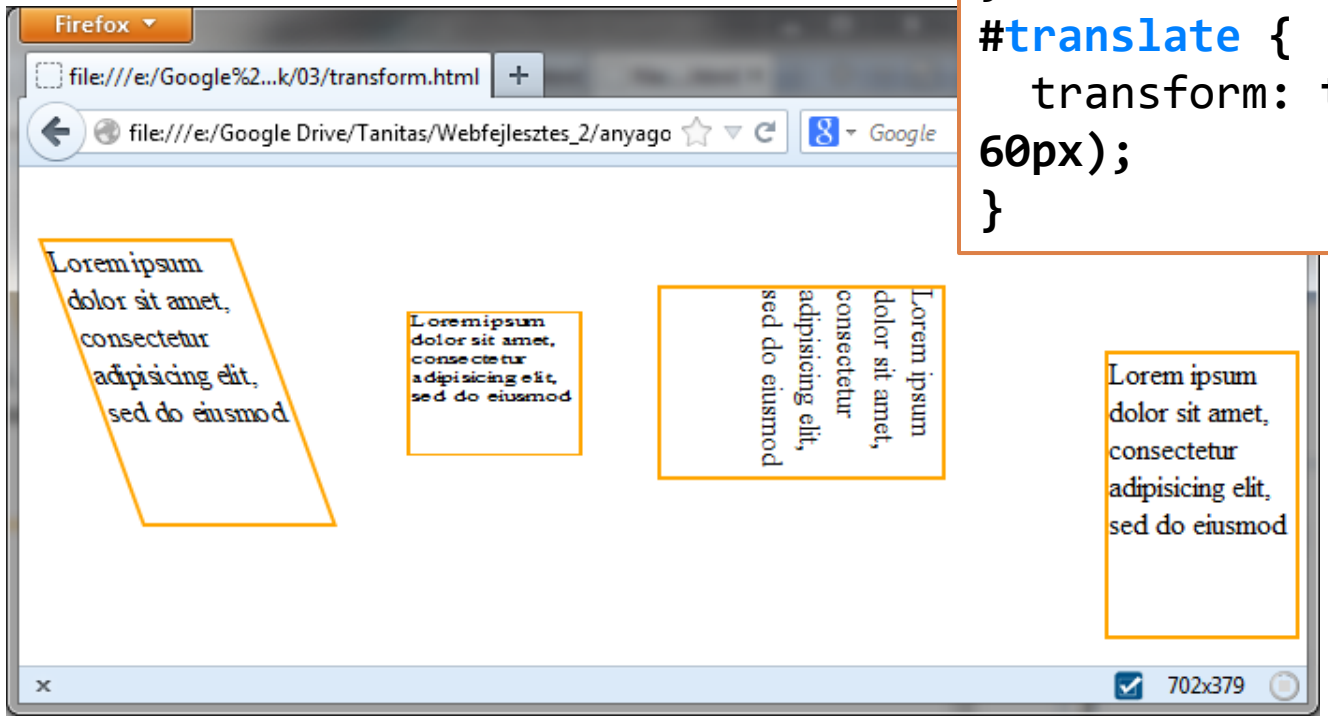
- Deklaratív módon
  - Transzformációk
  - Átmenetek
  - Animációk
- Böngészőtámogatottságuk egyre nagyobb
  - Mobil böngészők is
- Ha nem támogatják, akkor sincs baj, az ezzel járó többlet nincs kihasználva

CSS3 transzformációk

- Nem animáció
- Újabb stílustulajdonság: transform
- 2D transzformációk
  - skew, scale, rotate, translate
- 3D transzformációk
  - translate3d, scale3d, rotateX, rotateY, rotateZ

2D transzformációk


```
#skew {
  transform: skew(20deg);
}
#scale {
  transform: scale(0.9,0.5);
}
#rotate {
  transform: rotate(90deg);
}
#translate {
  transform: translate(50px,
60px);
}
```



3D transzformációk


```
#alap {
  perspective: 800px;
  perspective-origin: 50% 100px;
}
#scale {
  transform: scale3d(0.9,0.5,0.1);
}
#translate {
  transform: translate3d(50px, 60px, 100px);
}
#rotateX {
  transform: rotateX(45deg);
}
#rotateY {
  transform: rotateY(45deg);
}
#rotateZ {
  transform: rotateZ(45deg);
}
```

Átmenetek

- transition stílustulajdonság
- Az adott tulajdonság folytonos átmenettel megy egyik állapotból a másikba
- Böngésző végzi el, optimalizált módon
- Hatékonyabb
- Deklaratív
- Ha nem támogatott, akkor rögtön felveszi az értéket

Átmenetek

- transition részei
  - transition-property: melyik tulajdonság vagy all
  - transition-duration: mennyi idő alatt
  - transition-timing-function: milyen módon
  - transition-delay: mennyi idő múlva
- Vendor prefixek: -moz, -webkit, -ms, -o
  - pl. -webkit-transition
- Mindegyik tulajdonság animálása: all
- Nem mindegyik tulajdonságok animálható (lista)

Animáció – példa transition


```css
.doboz {
  width: 100px;
  height: 100px;
  border: 2px solid orange;
  transition: width 1s ease-in-out 0s;
}
```

```javascript
function atmenet() {
  var d = document.getElementById('doboz');
  d.style.width = '500px';
}
```

Animáció – példa transition


```
.doboz {
  width: 100px;
  height: 100px;
  border: 2px solid orange;
  transition: width 1s ease-in-out 0s;
}
.doboz:hover {
  width: 500px;
}
```

JavaScript és transition

- transition tulajdonság JavaScriptből is hozzáadható

```
elem.style.transition = 'all 1s linear 0s';
```

- Animáció vége esemény: transitionend

```
elem.addEventListener('transitionend', fuggveny, false);
```

CSS3 animáció

- Az átmenet két pont közötti egyszeri átmenetet tesz lehetővé, valamilyen matematikai elmozdulás-idő görbe alapján.
- CSS3 animáció
  - Két végállapot közötti állapotokat definiál (kulcskockák)
  - Ismétlődő végrehajtás

CSS3 animáció példa


```
@keyframes anim {
  0% {
    border: 1px solid orange;
  }
  50% {
    width: 200px;
    background-color: blue;
  }
  100% {
    border: 30px solid orange;
    background-color: green;
    width: 600px;
  }
}
/*...*/
.doboz:hover {
  animation-name: anim;
  animation-duration: 1s;
  animation-iteration-count: 4;
  animation-direction: alternate;
  animation-timing-function: ease-in-out;
}
```

Összefoglalás

- Nyelvi elemek és szolgáltatások
  - Névtelen függvények
  - Reguláris kifejezések
  - Időzítők
- Stílusok
  - style attribútum
  - class attribútum
  - stíluslapok
  - animációk