

Aktív HTML technológiák, JavaScript

Bakay Árpád

NETvisor kft

(30) 385 1711

arpad.bakay@netvisor.hu

Servlet, JSP, JSF - ne feledjük:

Ezek **szerver oldali** technológiák

A browser ablak (a user controlok felhasználó általi kitöltésén/beállításán túl) csak akkor változik, ha egy kéressel egy új lapot tölt le.

- Érezhető késleltetés, a hagyományos GUI-tól elmaradó élmény
- Felesleges teher a vonalakon és a szervereken.
- Mára már elavultnak érezzük a legtöbb ilyen interfészt.

Lássuk, miből mit főzhetünk a

kliens (browser) oldalon

HOT Web GUI Technológiák

Trendek

1. Javascript kiterjedt használata
2. Komplettn GUI logika implementálása JavaScript-ben
 - Egymással intenzíven kooperáló HTML elemek
 - UI library elemek alkalmazása
3. (Igazi) AJAX: a lap a teljes frissítése nélkül további adatokat cserél a szerverrel

Javascript

- Kb 15 éve létező és fejlődő, OO scripting nyelv
- Lényegében idegen a Java-tól
 - De legalább: C-szerű szintaxissal: keywordok, operátorok stb.
- Típus nélküli változók, amelyek skalár vagy objektum értékűek lehetnek
- Az Objektumok változtatható szerkezetű, egyszerű key-value tömbök!
 - Nincs igazi osztály-fogalom (legfeljebb prototípusok)

Javascript folyt.

A függvények

- Szintén objektumok: First Class Objects!
- closure-ként jól használhatók (pl. callback definiálása)
- minden függvény minden objektumon metódusként is hívható!!!
- variadikus: a deklarált és tényleges paraméterek száma nem kell egyezzen

A Closure

A függvények változókhöz rendelhetők, és ezzel nemcsak a függvény lokális állapota, de környezete is megmarad későbbre.

```
function createMsg(message) {  
    return function() {  
        return('The message I remember is ' + message);  
    };  
};  
  
MyVar1 = createMsg('First message');  
MyVar2 = createMsg('Second message');  
  
document.write(MyVar1());  
document.write(MyVar2());
```

Ez különösen jól felhasználható: callback-eknél, ahol egy függvény átadása annak egész környezetét is viszi magával.

Bővebben l. a [JavaScriptTutorial.html](#) dokumentumot!.

Javascript futása a browserben

- Amit a `<script>` tag tartalmaz, a lap betöltésekor lefuttatja.

```
<script>
```

- `document.write('Igy generálok contentet');`

```
</script>
```

- `src` attribútummal külső forrás is megadható

```
<script type="text/javascript" src="abc.js" />
```

- UI elemek `on-methodusai` javascriptet hívnak

```
<button onmouseover="alert('Ide ne!!!')" />
```

- További JS indító események: User-defined callback-ek
 - Pl. Timer és hálózati (letöltési) eseményeknél

Javascript példa

text output, objektum, függvény-literal

```
<script type="text/javascript">
  document.write('Hej!!');

  var myObj = {
    udvozet: "Hello, ",
    megmutat: function(kinek) {
      alert( this.udvozet + kinek + '!'); }
  }
  myObj.megmutat('World');
</script>
```

alert() : feldob egy üzenet ablakot, szöveggel és egy OK gombbal. Minden JavaScript környezetben.

document.write(): szöveget szúr be az aktuális HTML dokumentumba, a <script> tag helyére. Csak HTML környezetben futtatott Javascript-nál.

XX-ik századi Javascript divat

Elemek megjelenésének dinamikus változtatása

- Pl. animált menük, ha az egér fölé kerül
- Validálás
- Popup ablakok

```
<form onsubmit="javascript:return confirm('Send the query?');">
```

- Helyben generált HTML, pl. txt-ként letöltött adatok táblázatba rendezése.

A Javascript hatalma a browserben: a DOM-fa

- Egy objektum fa, amely a HTML elemek **pillanatnyi** állapotát tükrözi
- Olvasható és írható
- Azonnal frissül!
 - Pl. új elem berakása:

```
var olddiv = document.getElementById('szulo');  
  
var newdiv = document.createElement('div');  
  
newdiv.setAttribute('id','gyerek'); newdiv.innerHTML = 'This is the  
content <img src=„virag.jpg”/>';  
  
olddiv.appendChild(newdiv);
```

DOM alapok

- a „document” változó: a fa gyökere
- Az `id= attr`-al jelölt tag-ek különösen könnyen hozzáférhetők:

```
var myNode = document.getElementById('form1');
```

- Minden node-nak (element-nek) van `get/setAttribute()`: tag attribútumok
- `innerHTML()`: a tagon belüli content, string formában (írható is!)
- **Navigáció** `childNodes()`, `firstChild()`, `lastChild()`, `parentNode()`, `nextSibling()`, stb.
- Rendering info és widget működtetés, pl. `getClientTop()`, `myButton.click()`

Javascript események

Minden node-ot érhetnek események
Amelyekre callback függvényeket lehet
regisztrálni (itt is jól jönnek a closure-ok!)

- `onclick()`, `onfocus()`, `onmouseover()`

Propagation: minden eseményt általában megkapnak a szülő
nodeok is

Bővebben: <http://developer.mozilla.org/en/DOM>

Aszinkron hívások a szerver felé

Standard JavaScript API mechanizmus:

`XMLHttpRequest`

Aszinkron letöltést is támogat (és általában így használják):

```
xmlhttp = new XMLHttpRequest();  
xmlhttp.open("GET", „lista.txt”, true); // true = async!!  
xmlhttp.onreadystatechange( function(x) {  
    alert(xmlhttp.responseText);  
} )  
xmlhttp.send(null);
```

Mit lehet aszinkron leküldeni?

- Adatokat

Adatbázisból, röptében generálva, stb.

XML, CSV, v. JSON

- HTML/XML szegmenseket

XML adatok DOM fába parsolva is elérhetők a rsponseXML property-n keresztül

Az UI egyes területei egymástól függetlenül frissíthetők

PI: Portletek egy portálban

- Javascript kódot (eval mechanizmus)

„Generic computing engine”

JSON: JavaScript Object Notation

A Javascript objektum modelljéhez
jól igazodó szöveges serializálási formátum

```
{ „név”: „Ember Dániel”,  
  „lakcím”: „Teve u 3.”,  
  „gyerekek”: [  
    { „neve”: „Domi”,  
      „kora”: 17,  
      „fiu”: true,  
      „szulido”: „2005 szeptember 12” },  
    { „neve”: „Sára”,  
      „kora”: „16,,},  
    { „neve”: „Juli”,  
      „kora”: „14,,}  
  ]  
}
```

JSON adatok beolvasása: `var myObject = eval('(' + myJSONtext + ')');`
Vagy biztonságosabban: `var myObject = JSON.parse(myJSONtext, reviver)`

Kiírása: `var myJSONText = JSON.stringify(myObject, replacer);`

A CSS technológia

- Objektumok formátum-beállításai több, egyre specifikusabb rétegben definiálhatók
- Stílusok megadhatók
 - Osztályként a HTML headerben vagy egy külön .css fileban
 - Az objektum tag-jében a 'style' attribútummal
- Léteznek kényelmes „stílus próbálgató” webszolgáltatások, pl.

<http://www.spectrum-research.com/V2/generators/tableframe.asp>

Köszönöm a figyelmet!

arpad.bakay@netvisor.hu

Kérem olvassák el a JavaScript Tutorial-t is!

Ellenőrző kérdések

1. Foglalja össze a Javascript nyelv objektumainak sajátosságait!
2. Írja le, hogyan történik aszinkron kommunikáció a szerverrel
3. Mi a closure és hol használható?
4. Ismertesse röviden a browserek DOM fáját!
5. Ismertesse a JavaScript eseménykezelését
6. Milyen jellegzetességei vannak a Javascript függvényeknek?
7. Hol használhatunk függvény-literálokat?
8. Hogyan működik az XMLHttpRequest funkció?
9. Adjon rövid példát egy JSON dokumentumra, ami objektumokat és tömböket is tartalmaz.
10. Írjon 2-3 példát CSS selectorokra, és magyarázza el, mit specifikálnak.