TEXTAREA++

a JavaScript ereje

Bártházi András

- email: andras@barthazi.hu
- web: http://barthazi.hu
- tevékenységek:
  - Weblabor szerkesztő
  - NJSZT-WFSZ titkár
  - stb. :)

Fejlődő web

- A JavaScript a reneszánszát éli
- Diszkrét megoldások
- A következőkben a példák csak szemléltetőek, a praktikus, “jó” megoldásokat lásd Diszkrét JavaScript cikkemben: http://weblabor.hu/cikkek/diszkretjavascript

A textarea varázsa

- a textarea mint HTML elem
- a textarea és a JavaScript
- tippek a textarea használhatóbbá tételére
  - nem csak kezdőknek
  - nem csak haladóknak

a textarea mint HTML elem

- HTML elem
- <textarea></textarea> !
- id, class, lang
- name, rows, cols, disabled, readonly, accesskey, tabindex
- wrap

id

- egy *konkrét* elem beazonosítására
  - navigáció (weblap.html#id)
  - #id { tulajdonság: érték }
  - <label for="id">
  - document.getElementById('id')

class

- *hasonló célú* elemek besorolására (osztályozására)
  - .class { tulajdonság: érték }
  - egyszerre több osztály !
  - állapot ("open", "error")
  - getElementsByClassName(...)

lang

- a textarea tartalom nyelvének megadására
  - keresők (textarea esetén kérdés)
  - helyesírás ellenőrzők (SpellBound nem támogatja)
  - olvasható, használhatjuk JS-ből

name

- a textarea egy form elem, a name a POST/GET paraméter nevét határozza meg
  - értéke “CDATA”, gyakorlatilag ASCII karaktereket tartalmazhat
  - lehetőség: name=”form[text]”

rows, cols

- a textarea dimenzióinak meghatározására
  - nem teljesen pontos: IE OK, FF nem, Opera majdnem
  - CSS-ből felülbírálható

disabled="disabled"

- a szövegbevitel, form küldés tiltására
  - nem lesz elküldve, nem szerk.
  - IE: kijelölhető, fehér háttér
  - FF/Opera: nem jelölhető ki, szürke háttér

readonly="readonly"

- a szövegbevitel tiltására
  - nem szerkeszthető, form elemként elküldésre kerül
  - IE, FF, Opera: mintha "rendes" lenne, de nem módosítható

accesskey="U"

- egy gyorsbillentyű lenyomására fókuszba kerül az elem
  - IE és FF: Alt-U
  - Opera: nem támogatott?
  - hozzáférhetőséget javítja és rontja: a menüt nem lehet elérni

tabindex

- a tab billenytűvel történő "ugrálás" sorrendjét befolyásolja
- például a vizuálisan igen, de a HTML kódban nem egymás mellett levő elemekhez

wrap

- a textarea sortörésének beállítására
  - értékei: soft, hard és off lehetnek
  - nem szabványos!, de IE, FF és Opera is támogatja

a textarea és a JavaScript

DOM objektum

- JavaScriptből elérhető, programozható
- value, defaultValue
- style, blur(), focus(), setSelectionRange(), selectionStart, selectionEnd, Range

value, defaultValue

- a textarea aktuális és kezdeti tartalmának elérésére
  - a defaultValue is szabványos, IE, FF és Opera támogatja

.style

- az elem stílusjegyeinek állítására
  - elem.style.backgroundColor = ‘#ddd’;
  - elem.style.fontSize = ‘2em’;
  - elem.style.border = ‘1px solid #000’;
  - elem.
  - szabály: CSS “font-size” => JS “fontSize”

blur(), focus()

- metódusok a fókusz szabályozására
  - egy textarea-n meghívva a focus()-t, a fókusz rákerül
  - a blur()-t meghívva a fókuszt elveszti
  - oldal betöltődés: fókuszt a lényegre

setSelectionRange()

- a textarea-ban a kiválasztás (kurzor pozíció!) beállítására FF, Opera támogatja, IE nem
  - elem.setSelectionRange(2,4) => kiválasztás a 2. karaktertől a 4.-ig
  - elem.setSelectionRange(5,5) => kurzor 5. karakter utánra
  - nem árt egy focus(), mert azt nem befolyásolja

.selectionStart .selectionEnd

- a textarea-ban a kiválasztás / kurzor pozíciójának beállítására, kiolvasására - támogatás mint az előbb: IE nem, FF, Opera igen
  - var cursorpos = elem.selectionStart;
  - elem.selectionStart = elem.selectionEnd = 0;

TextRange

- IE által használt szövegtartomány típus
  - nincs közvetlen kapcsolatban a kurzor pozícióval
  - de a kiválasztásból, kurzor pozícióból szövegtartományt készíthetünk, illetve egy szövegtartomány által jelölt dokumentumrészt kiválaszthatunk

TextRange /2

- elem.setSelectionRange(5,10) utánzása:

```
var range = elem.createTextRange();
range.collapse(true);
range.moveStart('character', 5);
range.moveEnd('character', 10);
range.select();
```

TextRange /3

- “elem” kurzor pozíciójának lekérdezése (ronda, de megy)
  - var range = document.selection.createRange();

```
var length = elem.value.length;
var selectionLength = range.text.length;
var selectionStart =
  -1-range.moveStart(“character”, -length-1);
var selectionEnd = selectionStart + selectionLength;
```

Tippek

- Ennyi alapozás után jöjjenek a tippek
  - csak tippek, nem kidolgozott megoldások
  - várható egy letölthető anyag, amiben a konkrét példák is benne vannak (Weblabor.hu)

textarea++ / szavak száma

- szavak száma
  - mikor hasznos?
  - hogyan kell megszámolni a szavakat?
  - words = elem.value.split(/\s+/).length;
  - pl: window.status = words + ' szó';

textarea++ / becsukás

- az általában nem használt elemeket rejtsük el, mert csak a helyet foglalják
  - pl. szép url generálás, ha a gép jól ékezetmentesít...
  - prototype.js: Object.toggle(item);

textarea++ / max. hossz

- maximális hossz
  - nincs maxLength mint az <input>-nál
  - az adatbázis limitált, adatvesztés veszélye
  - azonnali visszajelzés
  - pl: onchange="if (this.value.length>1024) { alert ('hosszú'); this.value = this.value.substring(0,1024) }"

textarea++ / megváltozott

- vizuális visszajelzés, ha a textarea tartalma megváltozott
  - value és defaultValue összehasonlítása
    - textarea.changed { background: #eef; }
    - prototype.js: onkeyup="if (this.value!=this.defaultValue) { Object.addClassName(this, 'changed') } else { Object.removeClassName(this, 'changed' }"

textarea++ / betűméret

- a textarea-ban állítható betűméret:
  - nem jól látók, ergonómia
  - textarea.style.fontSize = ‘1em’; // IE-ben probléma
  - legyen külön CSS-ben, a’la oldal betűméret állítás - s mentsük is el hosszútávra

textarea++ / magasság

- a textarea magasságának állítása a felhasználói igények szerint
  - automatikus: tiszta megoldás nincs, de trükkök igen - a probléma: nem tudjuk, hány sor van a textarea-ban
  - kézi: két gomb, vagy drag'n'drop, utóbbi lásd Drupal 4.7
  - elem.style.height = (15*1.3) + 'em'; // 15 soros

textarea++ / HTML szerk.

- lásd Weblabor bbeditor.js
- eredetileg Publishing Kft.-nél (HA előadás) kifejlesztett megoldás
- a feladat: gombok, billentyűkombinációk felismerése, megfelelő jelölőelemek (pl: kirakása)

textarea++ / HTML szerk. 2

- 1) a kijelölés köré tegyünk ki egy nyitó és záró elemet
  - a) hol a kurzor? lásd korábban, posS, posE
  - b) mi van kijelölve? item.value.substring(posS, posE)
  - c) új tartalom összerakása: item.value eleje + nyitó + kijelölt + záró + vége

textarea++ / HTML szerk. 3

- 2) ha a kijelölés már tartalmazza a nyitó elemet, vagy záróelemet, akkor vegyük észre!
  - a) kijelölés tartalmát megszerezni, ahogy az előbb
  - b) regexp-et neki:

```
if (content.match(/<b>|<\/b>/)) { alert('b'); }
```

textarea++ / HTML szerk. 4

- ha találtunk, vegyük ki, s ne rakjunk be még
  - a legegyszerűbb reguláris kifejezéssel (replace) kivenni, a tartalom cseréje ugyanaz, mint a beszúráskor

textarea++ / HTML szerk. 5

- végül, de nem utolsó sorban: gondoskodni kell az új kurzor pozíciók, kiválasztás beállításáról is
  - a legkényelmesebb, ha az eredeti kijelölés megmarad, ha elé, mögé lett téve új elem, akkor az is legyen a kijelölés része
  - kijelölés mint a korábbiakban, a pozíció karakterlánc hosszakból számítható

Köszönöm

Kérdések?