Errata ke knize

JavaScript a Ajax

Moderní programování webových aplikací

Vážení čtenáři,

omluvte prosím problémy ve zdrojových kódech, jež vznikli lokaizací kódů do češtiny. Připravili jsme pro Vás errata, která chyby v knize opravují. Opravy byly zaneseny do zdrojových kódů, které jsou ke knize ke stažení.

redakce PC literatury nakladatelství a vydavatelství Computer Press

V celé knize:

- *Původně:* reference
- *Oprava:* ukazatel

Všude v knize:

- *Původně:* rámec
- *Oprava:* kontext

str. 14, správný kód:

```
// Vytvoříme nový objekt typu Rozvrh a uložíme jej do
// proměnné ‘mujRozvrh‘.
var mujRozvrh = new Rozvrh([
// Vytvoříme pole objektů typu Predmet, které je jediným
// parametrem konstruktoru objektů typu Rozvrh.
new Predmet("Tělocvik", "Novák"),
new Predmet("Matematika“, „Hliněný"),
new Predmet("AJ", "Procházková")
]);
// Zobrazíme informace o rozvrhu ve výstražném okně.
alert( mujRozvrh.zobraz() );
```

str. 15/odst. 5/ř. 2

- *Původně:* ukazatel <script> na ni

- *Oprava:* element script odkazující se na ni

str. 29/ 4/2

- *Původně:* umístění objektu

- *Oprava:* umístění objektu v paměti

str. 30, správný kód:

```
// Proměnná obj ukazuje na prázdný objekt.
var obj = new Object();
// objUkaz nyní ukazuje na stejný objekt.
var objUkaz = obj;
// Upravíme hodnotu nějakého atributu v původním objektu.
obj.nejakyAtribut = true;
// Vidíme, že změna se projevila v obou proměnných.
// (Jelikož obě ukazují na stejný objekt.)
alert( obj.nejakyAtribut === objUkaz.nejakyAtribut );
```

str. 33/1/1

- *Původně:* Protože JavaScript je dynamicky psaný jazyk

- *Oprava:* Protože je JavaScript jazykem skriptovacím,

str. 33/2/1

- *Původně:* Tento nástroj poskytuje

- *Oprava:* Tento operátor vrací

str. 33/2/2

- *Původně:* To může být perfektní řešení až na to, že proměnné typu objekt nebo pole, objekt jako třeba uživatel, vrací pouze objekt, čímž jsou obtížněji rozpoznatelné mezi všemi objekty.

- *Oprava:* To může být perfektní řešení s výjimkou proměnných typu objekt a pole. Pro objekt, např. Uzivatel, totiž operátor vrací řetězec object a nikoli přímo Uzivatel, čímž znemožňuje rozpoznatelnost mezi ostatními objekty.

Str. 42, správný kód:

```
// Jednoduchá funkce vezme jméno a uloží ho do příslušného atributu.
function Uzivatel( jmeno ) {
this.jmeno = jmeno;
}
```

Str. 45, správný kód:

```
// Konstruktor objektu reprezentujícího školní třídu.
function Trida( studenti, ucitel ) {
  // Soukromá metoda pro zobrazení všech studentů ve třídě.
  function zobraz() {
    alert( this.studenti.join(", ") );
  }

  // Uložíme data o třídě jako veřejné atributy objektu.
  this.studenti = studenti;
  this.ucitel = ucitel;

  // Zavoláme soukromou metodu pro zobrazení zprávy.
  zobraz();
}

// Vytvoříme nový objekt školní třídy.
var trida = new Trida( [ "Jan", "Bob" ], "Novák" );

// Volání selže, protože zobraz není veřejnou vlastností daného objektu.
trida.zobraz();
```

Str. 46, správný kód:

```
// Avšak můžeme zjistit hodnotu tohoto atributu pomocí metody vratjmeno(),
// která byla vytvořena dynamicky.
alert( uzivatel.vratjmeno() == "Bob" );
```

Str. 93, správný kód:

```
function prvni( elem ) {
elem = elem.firstChild;
return elem && elem.nodeType != 1 ?
dalsi ( elem ) : elem;
}
```

Str. 93, správný kód:

```
function posledni( elem ) {
elem = elem.lastChild;
return elem && elem.nodeType != 1 ?
pred ( elem ) : elem;
}
```

Str. 100, správný kód:

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>Zkouška načtení DOM</title>
  <script type="text/javascript" src="../Kapitola06/15-addEvent.js"></script>
  <script type="text/javascript" src="13-domready.js"></script>
  <script type="text/javascript">
    function tag(nazev, elem) {
      // Pokud nebyl předán element jako kontext, prohledáme celý dokument.
      return (elem || document).getElementsByTagName(nazev);
    }
    domReady(function() {
```

```
    alert( "DOM je načten!" );
    tag("h1")[0].style.border = "4px solid black";
  });
 </script>
</head>
<body>
 <h1>Zkouška načtení DOM</h1>
 <!-- Zde má být spousta HTML kódu. -->
</body>
</html>
```

Str. 106, správný kód:

```
// Prohlížeče, které nevychází z prohlížeče Mozilla:
strongElem.innerText
```

strana 109, správný kód:

```
function par(elem, nazev, hodnota) {
 // Ujistíme se, že byl předán správný název.
 if ( !nazev || nazev.constructor != String ) return '';

 // Zjistíme, jestli název odpovídá jednomu z podivných případů názvů.
 nazev = { 'for': 'htmlFor', 'class': 'className' }[nazev] || nazev;

 // Pokud uživatel nastavuje hodnotu
 if ( typeof hodnota != 'undefined' ) {
  // použijeme nejprve rychlý způsob.
  elem[nazev] = hodnota;

  // Pokud to lze, použijeme metodu setAttribute.
  if ( elem.setAttribute )
   elem.setAttribute(nazev,hodnota);
 }

 // Vrátíme hodnotu daného parametru.
```

```
  return elem[nazev] || elem.getAttribute(nazev) || ''; }
```

Str. 111, správný kód:

```
// Počkáme, až bude DOM připraven.
domReady(function(){

  // Najdeme všechny definice termínů.
  var dt = tag("dt");
  for ( var i = 0; i < dt.length; i++ ) {

    // Sledujeme událost klepnutí uživatele na daný termín.
    addEvent( dt[i], "click", function() {

      // Zkontroluje, jestli je příslušná definice již zobrazena.
      var zobrazena = par( this, "zobrazena" );

      // Zobrazíme/skryjeme definici.
      dalsi( this ).style.display = zobrazena ? 'none' : 'block';

      // Poznačíme si, zda je definice zobrazena.
      par( this, "zobrazena", zobrazena ? '' : 'ano' );
    });
  }
});
```

Str. 113, správný kód:

```
// Vytvoříme nový element <li>.
var li = vytvor("li");
par( li, "class", "novy" );

// Vytvoříme nový textový obsah a připojíme ho k elementu <li>.
pripoj ( li, "Děkujeme za návštěvu! " );
```

...

Str. 135, správný kód:

```
// addEvent/removeEvent napsal Dean Edwards, 2005
// doplnil Tino Zijdel
// http://dean.edwards.name/weblog/2005/10/add-event/

function addEvent(element, typ, obsluha) {
  // Každé funkci pro obsluhu události přiřadíme jedinečný identifikátor.
  if (!obsluha.$$jid) obsluha.$$jid = addEvent.jid++;

  // Vytvoříme hašovací tabulku typů událostí pro daný element.
  if (!element.udalosti) element.udalosti = {};

  // Vytvoříme hašovací tabulku funkcí pro obsluhu události pro každou
  // dvojici element/událost.
  var obsluhy = element.udalosti[typ];
  if (!obsluhy) {
    obsluhy = element.udalosti[typ] = {};

    // Uložíme funkci pro obsluhu události (pokud existuje).
    if (element["on" + typ]) {
      obsluhy[0] = element["on" + typ];
    }

  }

  // Uložíme funkci pro obsluhu události do hašovací tabulky.
  obsluhy[obsluha.$$jid] = obsluha;

  // Připojíme globální funkci pro obsluhu události, která odvede všechnu
  // práci.
  element["on" + typ] = zpracujUdalost;
};

// Počítadlo použité pro vytvoření jedinečného identifikátoru.
addEvent.jid = 1;
```

```
function removeEvent(element, typ, obsluha) {
  // Smažeme funkci pro obsluhu události z hašovací tabulky.
  if (element.udalosti && element.udalosti[typ]) {
    delete element.udalosti[typ][obsluha.$$jid];
  }
};

function zpracujUdalost(udalost) {
  var navratovaHodnota = true;

  // Získáme objekt události (IE používá globální objektu události).
  udalost = udalost || opravUdalost(window.event);

  // Získáme odkaz na hašovací tabulku funkcí pro obsluhu událostí.
  var obsluhy = this.udalosti[udalost.typ];

  // Spustíme všechny funkce pro obsluhu událostí.
  for (var i in obsluhy) {
    this.$$zpracujUdalost = obsluhy[i];
    if (this.$$zpracujUdalost(udalost) === false) {
      navratovaHodnota = false;
    }
  }

  return navratovaHodnota;
};

// Přidáme nějaké "chybějící" metody k objektu události v IE.
function opravUdalost(udalost) {
  // Přidáme standardní metody objektu události od W3C.
  udalost.preventDefault = opravUdalost.preventDefault;
  udalost.stopPropagation = opravUdalost.stopPropagation;
  return udalost;
};
```

```
opravUdalost.preventDefault = function() {
  this.navratovaHodnota = false;
};

opravUdalost.stopPropagation = function() {
  this.cancelBubble = true;
};
```

Str. 145, správný kód:

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  <style>p { height: 100px; }</style>
  <script src="01-ziskejStyl.js" type="text/javascript"></script>
  <script>
...
```

Str. 162/1/2

- *Původně:* uživatel právě komunikuje

- *Oprava:* uživatel právě pracuje

Str. 166, správný kód:

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  <title>DOM-Drag – ukázka přesunovatelného okna</title>
  <script src="22-DOMDrag.js" type="text/javascript"></script>
  <script type="text/javascript">
    window.onload = function(){
      // Inicializační funkce DOM-Drag, učiní element s parametrem
      // ID rovným 'okno' přesunovatelným.
      Drag.init( document.getElementById("okno") );
```

```
  };
 </script>
 <style>
  #okno {
   position: absolute;
   border: 1px solid #DDD;
   border-top: 15px solid #DDD;
   width: 250px;
   height: 250px;
  }
 </style>
</head>
<body>
 <h1>Ukázka přesunovatelného okna</h1>
 <div id="okno">Já jsem přesunovatelné okno, zkus mě někam
  přetáhnout!</div>
</body>
</html>
```

Str. 167, správný kód:

- viz. soubor 22-DOMDrag.js

Str. 173, správný kód:

```
<html>
<head>
 <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
 <title>script.aculo.us – ukázka přeuspořádání technikou
  táhni a pusť</title>
 <script src="lib/prototype.js" type="text/javascript"></script>
 <script src="lib/scriptaculous.js" type="text/javascript"></script>
 <script src="lib/effects.js" type="text/javascript"></script>
 <script src="lib/dragdrop.js" type="text/javascript"></script>
 <script type="text/javascript">
...
```

Str. 174, správný kód:

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
  <title>script.aculo.us – ukázka šoupátka</title>
  <script src="lib/prototype.js" type="text/javascript"></script>
  <script src="lib/scriptaculous.js" type="text/javascript"></script>
  <script src="lib/effects.js" type="text/javascript"></script>
  <script src="lib/dragdrop.js" type="text/javascript"></script>
  <script src="lib/controls.js" type="text/javascript"></script>
  <script type="text/javascript">
...
```

Str. 181, správný kód:

```
// Obecná funkce pro kontrolu, jestli je do vstupního prvku
// vložena nějaká informace.
function zkontrolujPovinne( elem ) {
    if ( elem.type == "checkbox" || elem.type == "radio" )
        return ziskejVstupyPodleJmena( elem.name ).pocetZaskrtnutych;
    else
        return elem.value || elem.value == elem.defaultValue;
}

// Najde všechny vstupní prvek s určeným jménem
// (vhodné pro hledání a práci se zaškrtávacími políčky a přepínači)
function ziskejVstupyPodleJmena( jmeno ) {
...
```

Str. 201, správný kód:

```
...
      // Zařídíme, aby byl při kliknutí zobrazen Lightbox.
      odkaz.onclick = function () {
          showLightbox(this);
```

```
        return false;
    };
...
```

Str. 207, správný kód:

```
...
// Najdeme všechny galerie na stránce.
var g = maTridu( “galerie”, “ul” );
...
```

Str. 209, správný kód:

```
...
// a skryjeme překrytí a galerii.
skryj( id(“prekryti”) );
skryj( id(“galerie”) );
...
```

Str. 209, správný kód:

```
...
// a zobrazíme ho pomocí efektu “fade in”.
roztmivej( prekryti, 50, 10 );
...
```

Str. 212, správný kód:

```
...
// a potom ji zobrazíme efektem “fade in”.
roztmivej( galerie, 100, 10 );
...
```

Str. 217, správný kód:

```
...
// A po třech a půl sekundách ho necháme ztmavnout.
setTimeout(function(){
ztmivej( galerie, 0, 10 );
...
```

Str. 225, správný kód:

```
...
// Navážeme spojení se serverem.
xml.send(null);
```

Str. 226, správný kód:

```
...
// Jiná množina dat s vícenásobnými hodnotami.
[
   { name: "jmeno", value: "Andrea" },
   { name: "prijmeni", value: "Peniaková" },
   { name: "jazyk", value: "JavaScript" },
   { name: "jazyk", value: "Perl" },
   { name: "jazyk", value: "Java" }
]
...
```

Str. 228, správný kód:

```
...
// Navážeme spojení se serverem.
xml.send(null);
```

Str. 229, výpis 10.6, správný kód:

```
...
```

```
// Navážeme spojení se serverem a pošleme data.
xml.send( "<polozky><polozka id='jedna'/><polozka id='dve'/></polozky>" );
...
```

Str. 230, správný kód:

```
...
// Otevřeme asynchronní požadavek typu GET
xml.open(„GET“, „/nejaka/url.cgi“, true);
...
```

Str. 230, správný kód:

```
...
// Navážeme spojení se serverem.
xml.send(null);
```

Str. 232, správný kód:

```
...
// Navážeme spojení se serverem.
xml.send(null);
```

Str. 234, správný kód:

- viz. soubor Kapitola10/11-ajax.js

- tento soubor je již v originálu bohužel špatně – autor naprosto zapomněl implementovat většinu z toho, o čem celou kapitolu mluvil

Str. 244, správný kód:

```
...
// Z každé zprávy v kanálu RSS získáme odkaz, titulek a popis.
var data = ziskejData( polozky[i] );
...
```

Str. 245, správný kód:

…

<item>

<title>Testovací zpráva</title>

<link>http://nejakaadresa.cz/?p=9</link>

<pubDate>Thu, 07 Sep 2007 09:58:07 +0000</pubDate>

<creator>Petra Osinová</creator>

<category>nezařazeno</category>

<description><![CDATA[Sem přijde obsah zprávy...]]></description>

</item>

…

Str. 247, správný kód:

…

// Žádáme jen o jednoduchou stránku, použijeme tedy GET.

typ: “GET“,

…

Str. 248, správný kód:

…

// Žádáme jen o jednoduchou stránku, použijeme tedy GET.

typ: “GET“,

…

Str. 248, správný kód:

…

// Umístíme novou položku do dokumentu.

obsah.appendChild(vytvorZpravu(polozky[i]));

…

Str. 249, správný kód:

…

```
// Získáme odkaz, titulek a popis z každé zprávy kanálu RSS.
var data = ziskejData( elem );
```

Str. 250, správný kód:

…

```
// Jednoduchá funkce pro získávání dat z elementu DOM.
function ziskejData( elem ) {
```

…

Str. 252, správný kód:

…

```
// Ukončíme procházení, narazíme-li na „starou“ zprávu
if ( ziskejData( polozky[i] ).link == posledniURL )
```

…

Str. 262, správný kód:

…

```
// znovu ho otevřeme - ale s prázdnou hodnotou
// (to řekne funkci otevreni(), že nemá znovu stahovat
// výsledky ze serveru, jen otevřít menu).
opt.otevreni( ‘‘, stav );
```

…

Str. 264, správný kód:

…

```
typ: „GET“,
url: “05-hledani.cgi?q=“ + posledniSlovo,
```

…

Str. 269, správný kód:

…

typ: „GET“,

url: “05-hledani.cgi?q=“ + posledniSlovo,

…

Str. 282, správný kód:

…

// Očekávaný formát vrácených dat je JSON.

dataType: „json“,

…