

11. JavaScript (3)

Obsah

- Zabránenie odoslaniu formulára
- Prehadzovanie obrázkov
- Funkcia eval()
- Dátum, čas

Zabránenie odoslaniu formulára

- Predchádzajúce formuláre sa odoslali vždy, aj keď neboli správne zadané údaje.
- Tomuto môžeme zabrániť nasledujúcimi spôsobmi:
 - vrátením hodnoty **false** pri udalosti **onclick** tlačidla typu **submit**
 - vrátením hodnoty **false** pri udalosti **onsubmit** formulára (elementu **<form>**).

Výsledný zdrojový kód by potom vyzeral nasledovne:

- `<input name="submit" type="submit" onclick="return spravne_udaje();" value="Odošli" />`
- `<form method="get" id="form1" onsubmit="return spravne_udaje();">`
- Funkcia `spravne_udaje()` musí vrátiť hodnotu `true` alebo `false`.

Zabránenie odoslania – ukážka

- Úprava ukážky 02 z predchádzajúcej prednášky.
- Pridáme booleovskú funkciu **odosli()**, ktorej hodnota bude návratovou hodnotou tlačidla Odošli (a teda celého formulára).

ukážka: ukazka-01.html

Prehadzovanie obrázkov

- Treba zmeniť zdroj obrázka – atribút **src**.
- 1. Ak máme obrázok pomenovaný pomocou atribútu **name**, môžeme použiť konštrukciu **document.meno_obrazka.src**.
- 2. Ak máme obrázok pomenovaný pomocou atribútu **id** prípadne pomocou atribútu **name**, môžeme použiť konštrukciu **document.images['id_obrazka'].src** alebo **document.getElementById('id_obrazka').src**

Prehadzovanie obrázkov – ukážky

- `document.images['velky'].src = 'obrazky/foto-01.jpg';`
- Rôzne spôsoby prehadzovania – pomocou `onclick`, `onmouseover`, odkaz, ...
- Ak do udalosti vložíme názov funkcie, **neuvádzame** kľúčové slovo **javascript:**, ale stačí napr. **`onclick="vymen()"`**. Jazyk funkcie je uvedený pri jej definícii.

ukážka: ukazka-02.html, ukazka-03.html, ukazka-04.html

Prehadzovanie – upozornenie

- Obrázky nie sú súčasťou formulára, aj keď sú vnútri elementu `<form>`.
- Ku zdroju obrázka **nemôžeme** pristupovať rovnako ako ku prvkom formulára: `document.form1.id_obrazka.src = cesta`.
- Každý veľký obrázok (okrem prvého) sa začne načítavať až v momente, keď sa zobrazuje, čo môže ovplyvniť rýchlosť zobrazovania obrázkov.

Ako vopred načítať obrázky

- Obrázky môžeme načítať do **cache (vyrovnávacej pamäte) prehliadača** a zároveň napr. do premenných (alebo poľa).
- Následne už nebudeme pracovať so zdrojovými súbormi (obrázkami), ale s premennými.
- Výhoda: všetky obrázky sa ihneď zobrazujú
- Nevýhoda: stránka sa dlhšie načítava (musia sa načítať všetky obrázky).

Objekt Image

- Štandardný objekt jazyka JavaScript
- Vytvárame inštanciu: obr = **new Image()**
- Po vytvorení má premenná všetky vlastnosti ako obrázok webovej stránky (**src**, alt, id, name, width, height...)
- Následne musíme do vlastnosti **src** priradiť zdroj obrázka: obr.**src** = 'foto-01.jpg'.

ukážka: ukazka-05.html

Zmena ukážky

- Doteraz sme menili len veľký obrázok. Ukážku zmeníme tak, že budeme meniť všetky malé obrázky.
- Každý (v dvoch verziách) si uložíme do samostatných premenných (**obr1_on**, **obr1_off**, ...).
- Pri nadídení ponad obrázok zobrazíme verziu **_on** a pri odídení z obrázka zobrazíme verziu **_off**.

Funkcia zobraz()

- Vytvoríme univerzálnu funkciu **zobraz()**, ktorá zobrazí správnu verziu obrázka.
- Funkcia dostane 2 parametre: **id_obrázka** a **premennú s obrázkom**, ktorú treba zobraziť.
- onmouseover="zobraz('obr1', 'obr1_on')"
- onmouseout="zobraz('obr1', 'obr1_off')"

Funkcia zobraz() – pokr.

- `function zobraz(id_obrazka, premenna) { }`
- **Nemôžeme použiť zápis**
`document.getElementById(id).src=obr1.src`
- Aj keď je v premennej obr1 obrázok (objekt Image), príkaz nebude fungovať. Musíme **donútiť** JavaScript, aby ho vyhodnotil -> použijeme funkciu **eval()**.

Štandardná funkcia eval()

- Vyhodnotí vstup
- V našom prípade potrebujeme vyhodnotiť výraz `obr1.src`. Ale musíme ho zapísať **eval(obr1 + ".src")**.
- Výsledný príkaz:
`document.getElementById(id_obrazka).src = eval(premenna + ".src");`

ukážka: ukazka-06.html

Dátum, čas

- Pred prácou s dátumom alebo časom najprv musíme vytvoriť premennú a priradiť do nej príslušný dátum, resp. čas.
- Napríklad `akt_datum = new Date();`
(získame aktuálny dátum a čas)
- S premennou `akt_datum` potom pracujeme pomocou nasledujúcich funkcií.

new Date()

- Príkaz `Date()` môžeme spustiť aj s niekoľkými parametrami:
- `new Date("Month dd, yyyy hh:mm:ss")`
- `new Date("Month dd, yyyy")`
- `new Date(yy,mm,dd,hh,mm,ss)`
- `new Date(yy,mm,dd)`
- `new Date(milliseconds)`

Funkcie na prácu s dátumom a časom

funkcia	popis
getDate()	deň v mesiaci, 1 – 31
getDay()	deň v týždni, 0 – 6 (0 = nedeľa, 1 = pondelok)
getMonth()	mesiac, 0 – 11 (0 = január, 1 = február)
getFullYear()	rok ako 4 číslice
getHours()	hodina, 0 – 23
getMinutes()	minúty, 0 – 59
getSeconds()	sekundy, 0 – 59
getMilliseconds()	milisekundy, 0 – 999

Ku takmer všetkým funkciám get* existujú aj funkcie set*, ktoré nastavujú príslušné zložky dátumu a času.

Aktualizácia dokumentu, vypisovanie do stránky

- `document.lastModified` – **vlastnosť**, ktorá vráti dátum a čas poslednej aktualizácie aktuálneho dokumentu (korektne funguje len pri statických stránkach).
- `document.write(str)` – funkcia, ktorá vypíše na stránku zadaný reťazec str.

ukážka: [ukazka-07.html](#)

**Ďakujem za
pozornosť 😊**

Priestor na Vaše otázky