# Úvod do jQuery

Tibor Kádek

tibor.kadek@gmail.com

# jQuery

- framework pre javascript
- umožňuje jednoduché vyhľadávanie elementov DOMu, ich modifikáciu a vytváranie nových
- práca s udalosťami
- manipulácia s CSS
- preddefinované funkcie efektov aj skladanie vlastných animácií
- pokročilé funkcie pre prácu s poliami
- podpora AJAX
- podpora pluginov
- pomocné utility atď.

# Spustenie kódu

- v hlavičke dokumentu

```
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
    $(document).ready(function(){
      // DOM* bol vytvorený a je pripravený
      // sem príde jQuery kód
    });
</script>
```

Pozn.: $ je alias pre jQuery()

* Document Object Model

# Práca s kódom

```
$("selektor na výber prvkov").metóda("parametre");
```

- výber DOM prvkov pomocou jQuery selektoru
  - **$** je alias pre triedu jQuery a teda **$()** vytvorí nový jQuery objekt, s ktorým budeme ďalej pracovať
  - vráti objekt typu jQuery

- väčšina metód takto vytvoreného jQuery objektu vráti tiež jQuery objekt a tak je možné reťazenie

```
$("prvky").metóda("parametre").metóda2("param");
```

# Výber elementov – Selektory

```
<div id="svet" class="pozdrav">Ahoj svet!</div>
```

- výber jedného prvku s ID "svet"

```
$("#svet");
```

- výber prvkov triedy "pozdrav"

```
$(".pozdrav");
```

- výber všetkých prvkov určitého typu

```
$("div");
```

# Selektory

- podpora CSS3 selektorov
- výber prvého (`:first`) a posledného (`:last`) prvku

```
$("tr:first"); // vráti prvý riadok tabuľky
$("tr:last"); // vráti posledný riadok tabuľky
```

- **:first-child** – výber prvého potomka
  - ekvivalentné s **:nth-child(1)**
  - narozdiel od `:first` môže vrátiť viac prvkov

```
$("li:first-child");
// ak existuje viac zoznamov, vráti viac prvkov
```

# Selektory

- výber párneho (`:even`) a nepárneho (`:odd`) prvku

```
<table>
        <tr><td>Riadok 0</td></tr>
        <tr><td>Riadok 1</td></tr>
        <tr><td>Riadok 2</td></tr>
        <tr><td>Riadok 3</td></tr>
</table>
$("tr:odd").addClass("zelena");
$("tr:even").addClass("cervena");
```

# Selektory

- možnosť filtrovať podla rôznych vlastností
- **:eq(index)** - zo zoznamu prvkov vyberie n-tý prvok, pričom sa pole indexuje od 0.

  Pozn.: Pri :nth-child(n) sa indexuje od 1
- **:lt(index)** - "menej ako" index
- **:gt(index)** - "viac ako" index

```
$("li:eq(4)");
$("li:lt(4)"); //vyberie prvky od 0 po 3
$("li:gt(4)"); //vyberie prvky od 5 a vyššie
```

# Selektory

- filtrovanie podľa viditeľnosti

  **:visible** – viditeľný prvok

  **:hidden** – skrytý prvok

- **:contains("text")** – vyberie všetky elementy, ktoré obsahujú zadaný text (case sensitive)

```
<p>John Michael Talbot</p>
<p>Kandy Talbot</p>
<p>Jozef Mrkva</p>
$("p:cointains('Talbot')").css('color', 'red');
// červené budú prvé dva odstavce
```

# Selektory

- **:empty** – vyberie elementy, ktoré neobsahujú potomka (vrátane textového elementu)

```
<table>
    <tr><td>text</td><td></td></tr>
    <tr><td><p></p></td><td></td></tr>
</table>
```

```
$("td:empty").text("bol prázdny");
// len pre pravé stĺpce doplní text "bol prázdny"
```

# Selektory

- **:animated** – vyberie elementy, ktoré sú práve v procese animácie
  - na zvýšenie výkonu sa odporúča najprv spraviť čistý CSS výber na ktorý sa použije konštrukcia **.filter(":animated")**
- **:disabled** – vyberie všetky elementy, ktoré sú blokované
- **:enabled** – vyberie všetky elementy, ktoré sú aktivované
  - odporúča sa používať s názvom tagu alebo iným selektorom, ináč sa použije univerzálny selektor "*"

```
<input name="email" disabled="disabled" />
$("input:disabled)");
```

# Selektory

- **:has(selektor)** – vyberie elementy, ktoré obsahujú aspoň jeden element, ktorý zodpovedá zadanému selektoru
  - keďže nie je súčasťou CSS špecifikácie, na zvýšenie výkonu sa odporúča použiť zreťazenie metódou **.has()**
- **:header** – vyberie všetky elementy, ktoré sú typu nadpis (h1, h2, h3 atď.
  - na zvýšenie výkonu sa odporúča najprv spraviť čistý CSS výber na ktorý sa použije konštrukcia **.filter(":header")**
- **:not(selektor)** – negácia výberu – vyberie elementy, ktoré nezodpovedajú filtru

```
$("input:not(:checked)");
// vyberie tie, ktoré nie sú označené
```

  - pre zložitejšie konštrukcie sa odporúča kvôli čitateľnosti zápisu použiť zreťazenie výberu metódou **.not()**

# Selektory

- **selektory na vyhľadanie formulárových prvkov** – na zvýšenie výkonu v moderných prehliadačoch sa používa ekvivalentný zápis [type="prvok"]

  **:button** – najprv spraviť čistý CSS výber, na ktorý sa použije **.filter(":button")**

  **:checkbox –** ekvivalentné s **[type="checkbox"]**

  **:file** – ekvivalentné s **[type="file"]**

  **:image** – ekvivalentné s **[type="image"]**

  **:input** – najprv spraviť čistý CSS výber, na ktorý sa použije **.filter(":input")**

  **:password** – ekvivalentné s **[type="password"]**

  **:radio** – ekvivalentné s **[type="radio"]**

  **:reset** – ekvivalentné s **[type="reset"]**

  **:submit** – ekvivalentné s **[type="submit"]**

  **:text** – ekvivalentné s [type="text"]

# Manipulácia s DOM elementami

- **.attr("názov atribútu")** - metóda vráti hodnotu atribútu zvoleného elementu (ak zoznam obsahuje viac elementov, vráti hodnotu prvého)
  - ak chceme získať hodnotu každého elementu zo zoznamu, je potrebné použiť konštrukciu cyklu pomocou metódy ***.each*** alebo ***.map***

```
<em title="jednoduchý framework">jQuery</em>
```

```
var title = $("em").attr("title");
// title obsahuje jednoduchý framework
```

# Manipulácia s DOM elementami

- **.attr("názov atribútu", "hodnota")** - nastaví zvolenému elementu hodnotu atribútu

  **.attr(mapa dvojíc typu atribút: "hodnota")**

- **.removeAttr("názov") - odstráni atribút každému zo zvolených elementov**

```
$('#photo').attr('alt', 'západ slnka');


$('#photo').attr({
    alt: 'západ slnka',
    title: 'photo CC-BY Kandy Talbot'
$});
```

# Manipulácia s DOM elementami

- **.html()** - získa HTML obsah z prvého elementu zo zoznamu zvolených elementov
  - používa *innerHTML* prehliadačov
- **.html("HTML reťazec")** - nastaví zvoleným elementom obsah

```
<div></div>
<div><p>tento paragraf sa tiež nahradí</p></div>
<div></div>
```

```
$('div').html('<p><em>HTML</em> reťazec</p>');
```

# Manipulácia s DOM elementami

- **.text()** - získa kombináciu textového obsahu zo všetkých zvolených elementov a ich potomkov (odstráni značkovanie)
  - narozdiel od *.html()* metódy je metódu *.text()* možné použiť aj na XML dokument
  - na formulárové prvky treba použiť metódu *.val()*
- **.text("reťazec")** - nastaví zvoleným elementom textový obsah
  - špeciálne znaky sa nahrádzajú HTML entitami (napr. < pre <)

```
<div><p><em>testovací</em> paragraf</p></div>
```

```
var str = $("div:first").text();
// str obsahuje reťazec "testovací paragraf"
```

# Manipulácia s DOM elementami

- **.val()** – vráti hodnotu prvého elementu zo zoznamu zvolených elementov
  - primárne sa používa pre formulárové prvky *input*, *select*, *textarea*
  - ak sa jedná o <*select multiple="multiple"*>, vráti pole obsahujúce všetky vyznačené možnosti; ak nieje vyznačená žiadna možnosť, vráti null

```
$('select.foo option:selected').val();  $('select.foo').val();
// získa hodnotu vybranej možnosti roletového menu

<select id="multiple" multiple="multiple">
    <option selected="selected">Multiple</option>
    <option selected="selected">Multiple2</option>
</select>
var multipleValues = $("#multiple").val();
// multipleValues je pole reťazcov ['Multiple', 'Multiple2']
```

# Manipulácia s DOM elementami

- **.val("hodnota")** – nastaví hodnotu zvoleným elementom
  - vstupom môže byť pole reťazcov, ktoré korešpondujúce elementy nastaví ako označené (selected/checked) a ostatné elementy budú odznačené

```
<select id="multiple" multiple="multiple">
    <option>Multiple</option>
    <option>Multiple2</option>
    <option selected="selected">Multiple3</option>
</select>
<input type="checkbox" name="ch" value="check1"/> check1
<input type="checkbox" name="ch" value="check2"/> check1
$('#multiple').val(["Multiple", "Multiple2"]);
$('input').val(["check1", "check2"]);
// budú označené všetky elementy okrem Multiple3
```

# Manipulácia s CSS

- **.addClass("názov triedy")** – pridá (nenahrádza) triedu/triedy všetkým vybraným elementom
  - je možné pridať viac tried súčasne - stačí ich oddeliť medzerou

```
$("p").addClass("trieda1 trieda2");
```

- **.removeClass("trieda")** – odstráni triedu zo všetkých vybraných elementov
- často sa používa prepínanie z jednej triedy na druhú

```
$("p").removeClass("prva").addClass("druha");
```

# Manipulácia s CSS

- **.css("názov vlastnosti")** – vráti hodnotu CSS vlastnosti prvého elementu zo zoznamu vybraných
  - združené vlastnosti nie sú povolené (margin, border...)
  - vie rovnako interpretovať CSS aj DOM formátovanie viacslovných vlastností (background-color, backgroundColor)

```
var color = $("div").css("background-color");
```

- rôzne prehliadače môžu vrátiť hodnoty farieb rôzne – logicky sú síce rovnaké, ale textovo nie (#FFF, #ffffff, rgb(255,255,255))

# Manipulácia s CSS

- **.css("vlastnosť", "hodnota")** – nastaví vybraným elementom požadované vlastnosti
- **.css(mapa dvojíc typu "kľúč": "hodnota")**
- od jQuery 1.6 pribudla podpora relatívnych hodnôt na inkrementáciu (**+=**) a dekrementáciu (**-=**) aktuálnej hodnoty

```
$('#box').css('width', '+=200');
$('#box').css({
    'color': 'green',
    'background-color': '#fff'
});
```

# Manipulácia s CSS

- ak potrebujeme *Integer* hodnoty na matematické výpočty, použijeme nasledujúce metódy, ktoré narozdiel od metódy .css("vlastnosť") vrátia hodnotu bez 'px'
- **.width() / .height()** – vypočíta aktuálnu šírku/výšku elementu
- **.innerWidth() / .innerHeight() –** vypočíta aktuálnu šírku/výšku elementu vrátane padding ale bez border
- **.outerWidth() / .outerHeight() –** vypočíta aktuálnu šírku/výšku elementu vrátane padding aj border a ak je vstupným parametrom *true*, tak aj margin

# Efekty

- **.hide()** – schová zvolené elementy
  - ekvivalentné s volaním **.css('display', 'none')**

**.hide([duration] [, easing] [, callback])**

  - *duration* – reťazec alebo číslo v milisekundách, ktoré udáva, ako dlho bude animácia trvať
  - reťazec **'fast'** má hodnotu 600 a **'slow'** 200 milisekúnd
  - *easing* udáva spôsob, akou rýchlosťou animácia prebieha v rozličných miestach animácie – implicitne má hodnotu **'swing'** a pre konštantný priebeh **'linear'**

```
$('#box').hide('slow', function() {
    alert('Animácia skončila.');
});
```

# Efekty

- **.show()** – zobrazí zvolené elementy
  - ekvivalentné s volaním **.css('display', 'block')**

  **.show([duration] [, easing] [, callback])** – pozri hide()
  - ak je v štýle použité !important, je potrebné použiť *.css('display', 'block !important')*
- **.toggle([duration] [, easing] [, callback])**
  - ak sú zvolené elementy zobrazené, schová ich; ak schované, zobrazí ich

```
<p style="display: none">Toggle</p>
$('p').toggle(); // zobrazí
$('p').toggle(); // schová
```

# Efekty

- **.fadeIn([duration] [, easing] [, callback])** – zobrazí zvolené elementy postupným znepriehľadňovaním (zmenou opacity)
- **.fadeOut([duration] [, easing] [, callback])** – schová zvolené elementy postupným spriehľadňovaním
- **.fadeToggle([duration] [, easing] [, callback])**
  - ako pri *.toggle()* len sa animuje priehľadnosť
- **.fadeTo(duration, opacity [, easing] [, callback])**
  - **opacity** – číslo medzi 0 a 1 určujúce výslednú priehľadnosť

*fadeTo(0, opacity)* je ekvivalentné s *.css('opacity', opacity)*

```
$("#box").fadeTo(500, 0.5);
// zmení priehľadnosť z pôvodnej opacity na 50%
```

# Udalosti / Events

- metódy, ktoré zachytávajú správanie používateľa pri práci s prehliadačom a môžu následne manipulovať s registrovanými udalosťami
- parametrom metód je odkaz na funkciu, ktorej sa predá objekt typu event a funkcia sa spustí pri každom vyvolaní udalosti
    - často sa vytvorí len anonymná funkcia

```
function(event){}
```

- je možné zabrániť implicitnému správaniu s **event.preventDefault()**

# Udalosti / Events

- **.click( handler(eventObject) )** – reaguje na udalosť kliknutia myšou na vybrané elementy
  - spracuje sa, len ak nastane séria nasledovných udalostí
    - tlačidlo myši je stlačené, kým je v priestore elementu
    - tlačidlo myši je pustené, kým je v priestore elementu
  - ak je požadované iné chovanie, môže sa použiť **.mousedown()** a **.mouseup()**

```
$("a").click(function(event){
        alert("Používateľ klikol na odkaz.");
        event.preventDefault();
  });
// po kliknutí na odkaz sa zobrazí hláška, ale neaktivuje sa odkaz
```

# Udalosti / Events

- **.hover(handlerIn(eventObject), handlerOut(eventObject))**

  – reaguje na udalosť, keď používateľ prejde myšou nad elementy a opustí ich

```
$("td").hover(
  function () {
    $(this).addClass("hover");
  },
  function () {
    $(this).removeClass("hover");
  }
);
```

# Udalosti / Events

- **.focus()** – reagujú najme na udalosti formulárových prvkov – napr. keď klikneme do vstupného pola (input), alebo vyberáme z roletového menu (select)
- **.blur()** – udalosť sa pošle, ak element stratí focus
- **.change()** – reaguje na zmenu hodnoty zvoleného elementu
  - použitie je limitované na formulárové prvky input, textarea, select
- **.keydown(), .keyup()** – práca so vstupom z klávesnice
  - udalosť sa pošle len elementu, na ktorom je focus

# AJAX

- načítanie dát zo servera bez nutnosti znovunačítať celú stránku
- **$.load(url)** – najjednoduchší spôsob, ako načítať html kód zo servera do vybraného elementu

```
$('#obsah').load('ajax/test.html');
```

- **$.get(url [, dáta] [, callback funkcia pri úspešnom načítaní] [, "typ dát"])**

```
$.get("test.php",
    function(data){
      $('body').append( "Name: " + data.name ) // John
               .append( "Time: " + data.time ); //  2pm
    }, "json");
// test.php vráti údaje v json formáte
// echo json_encode(array("name"=>"John","time"=>"2pm"));
```

# AJAX

- **$.post(url [, dáta] [, callback funkcia pri úspešnom načítaní] [, "typ dát"])**
  - dáta vo forme mapy reťazcov sa pošlú ako POST request

```
$.post("test.php", { name: "John", time: "2pm" },
   function(data){
     $('body').append( "Name: " + data.name )
              .append( "Time: " + data.time );
   }, "json");
// echo json_encode(array("name"=>$_POST['name'],
"time"=>$_POST['time']));
```

# Tvorba vlastných pluginov

- pridanie vlastnej metódy ako rozšírenie jQuery objektu s implicitnými hodnotami vlastností

```
jQuery.fn.priklad = function(options) {
    var settings = jQuery.extend({
        value: 5, name: "pete", bar: 655
    }, options);
    // sem príde kód metódy
};
```

- metóda **$.extend()** je utilitou jQuery, ktorá zlúči dva objekty modifikáciou prvého