Publikace vznikla v rámci projektu OPVK „Vyškolený pedagog – záruka kvalitní výuky“ na Střední odborné škole veterinární, mechanizační a zahradnické a Jazykové škole s právem státní jazykové zkoušky v Českých Budějovicích, reg. č. CZ.1.07/1.3.40/01.0007.

Ing. Roman Blábolil
České Budějovice, 2013
Jazyková korektura: Neprošlo jazykovou korekturou

Sazba: Ing. Roman Blábolil

© Ing. Roman Blábolil
OBSAH

1 Úvod .................................................................................................................................................. 5
2 AJAX (Asynchronous JavaScript and XML) ...................................................................................... 6
  2.1 AJAX ............................................................................................................................................. 6
  2.2 AJAX vs. AJAJ ............................................................................................................................. 7
  2.3 AJAJ/AJAJ v praxi ....................................................................................................................... 7
3 Protokoly a adresy ............................................................................................................................... 9
4 Jazyk HTML a XHTML ...................................................................................................................... 14
5 Kaskádové styly - CSS ..................................................................................................................... 18
  5.1 Třídy ............................................................................................................................................. 21
  5.2 Nejpoužívanější atributy CSS stylů ............................................................................................ 26
5 JavaScript .......................................................................................................................................... 37
  5.1 Charakteristiky jazyka JavaScript ............................................................................................... 37
  5.2 Nejčastější aplikace psané v JavaScriptu ..................................................................................... 38
  5.3 Schéma komunikace PHP ........................................................................................................... 39
  5.4 Způsob zápisu JavaScriptu do dokumentu ..................................................................................... 39
  5.5 Zápis do proudu dokumentu .......................................................................................................... 40
  5.6 Externí zápis JavaScriptu ............................................................................................................... 41
  5.7 In-line zápis JavaScriptu ............................................................................................................... 42
  5.8 Kombinovaný zápis JavaScriptu ..................................................................................................... 43
  5.9 Příkazy jazyka JavaScriptu ............................................................................................................ 45
  5.10 Proměnné, datové typy, operátory ............................................................................................... 46
    5.10.1 Proměnné JavaScriptu ............................................................................................................ 46
    5.10.2 Datové typy JavaScriptu ........................................................................................................ 47
    5.10.3 Operátoré JavaScriptu ............................................................................................................ 49
5.11 Hlášky v JavaScriptu ..................................................................................................................... 50
    5.11.1 Alert ....................................................................................................................................... 51

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ
5.11.2 Prompt.............................................51
5.11.3 Confirm........................................52
5.12 Řídící struktury v JavaScriptu.................................53
  5.12.1 If..............................................53
  5.12.2 While ........................................55
  5.12.3 For .............................................55
  5.12.4 Switch........................................56
5.13 Řízení běhu programu ........................................57
5.14 Skripty reagující na událost.................................58
  5.14.1 Události myši..................................58
  5.14.2 Události stránky a okna..............................60
5.15 Funkce ..............................................61
5.16 Pole....................................................61
  5.16.1 Asociativní pole........................................63
5.17 Objetkový model JavaScriptu.................................63
5.18 Datum a čas.............................................66
6 AJAX – základní užití .............................................67
  6.1 Požadavek typu POST ................................68
  6.2 Objekt XMLHttpRequest ................................68
  6.3 Odpověď serveru ......................................69
  6.4 Onreadystatechange událost..............................70
7 AJAX a Javascriptové frameworky...............................71
  7.1 Ukázka jednoduchého AJAX chatu – autor Míka M........71
8 Seznam použité literatury........................................74
1 ÚVOD

Tato publikace vznikla v rámci projektu OPVK „Vyškolený pedagog – záruka kvalitní výuky“. Cílem publikace je zpracovat základní poznatky o fungování technologie AJAX.

Publikace se bude zabývat těmito tématy:

Základní přehled a principy technologií pro AJAX

- protokol HTTP
- jazyk HTML a XHTML, kaskádové styly CSS
- jazyk JavaScript
- objektový model JavaScriptu
- jazyk XML, JSON
- techniky programování na straně serveru PHP, ASP.NET

Jak funguje AJAX

- jednoduchá WWW aplikace s použitím AJAXu a popis funkce
- Bezpečnost
- Ověřování dat ve formulářích
2 AJAX (ASYNCHRONOUS JAVASCRIPT AND XML)

Ajax není nový programovací jazyk ale nový způsob programování.

Je to základní technologie pro aplikace, které potřebují dynamicky měnit obsah bez znovu načtení stránky prohlížeče. Jak rozepsaný název napovídá, využívá programovací jazyk JavaScript.

V praxi se používá při tvorbě www stránek kombinace různých technologií. Většinou se jedná o:

- HTML nebo XHTML
- CSS
- PHP
- JavaScript

2.1 AJAX

- Je technika na straně klienta (webového prohlížeče), která pomáhá vytvářet asynchronní aplikace.
- AJAX je zodpovědný za odeslání požadavku a předání zpětné vazby klientovi, který jí následně může a nemusí zpracovat.
- AJAX vychází z pojmu XML, tedy práci s XML ale v dnešní době se už moc nepoužívá a byl nahrazen formátem JSON (AJAJ).
2.2 AJAX vs. AJAJ

Formát XML

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<root>
  <element>Test</element>
  <element2>Test 2</element2>
</root>
```

Formát JSON

```json
dsdict
{"root":{"element":"Test","element2":"Test2"}}
```

Formát JSON je rychlejší na psaní a dosahuje mnohem vyšší “komprimace” dat.
Formát JSON nezatěžuje tolik server a síť.

2.3 AJAX/AJAJ V PRAXI

Asynchronní volání dnes lze vidět všude.

- Nejznámější je klasické vyhledávání v Google (návrh frází při vyhledávání)

Zděné pochopení je vyhledávání požadavek na webový server a žádá od něho návrhy možných vyhledávacích frází. AJAJ požadavek zabezpečí, že prohlížeč klienta dostane data. Po přijetí výsledku z webového serveru, připravená funkce zpracuje...
výsledek a ukáže klientovi návrhy níže pod vstupním polem. Toto vše se děje bez znovu načtení stránky a to díky technologii AJAX.

- Facebook aplikace
- Velká řada webových e-mailových klientů

V případě otevřené konzole webového prohlížeče lze sledovat, zda ze stránky odchází asynchronní požadavek.
3 PROTOKOLY A ADRESY


Nejpoužívanější aplikační protokoly jsou:

HTTP (Hyper Text Transfer Protocol) – přenos stránek systému World Wide Web. Tento protokol umí pracovat s tzv. hypertextovými daty, což jsou data, která obsahují text, obrázky, video, zvuk a podobně.

SMTP (Simple Mail Tranfer Protocol) – protokol pro přenos elektronické pošty.

POP3 (Post Office Protocol) -protokol k přijímání elektronické pošty poštovním klientem.

FTP (File Transfer Protocol) – protokol pro přenos souborů.

Telnet (Telecommunication network) – protokol pro přístup ke vzdáleným počítačům.

Tyto protokoly používají ke komunikaci tzv. systémové porty, jaké „brány“ do počítače. Je jich teoreticky 65535. Každá služba používá jiný port

(web - 80, odchozí pošta - 25 atd.)

Program, který odděluje počítač od internetu a povoluje definované služby a porty se nazývá FIREWALL.
DNS – adresa počítače


Vazba mezi jménem počítače a IP-adresou je definována v DNS (Domain Name System) databázi, která je celosvětově distribuovaná. To znamená, že například doménové jméno soscb.cz bude známé jak V České republice tak třeba i v Mexiku.

Před navázáním spojení musí počítač přeložit jméno na IP-adresu:

1. Přelož www.soscb.cz na IP-adresu
2. IP-adresa je 90.183.11.178
3. navázání spojení

Obr. 1

Pokud by jste chtěli zjistit přes jaké servery putují vaše data, můžete použít příkaz tracert napsáním do příkazového řádku: tracert www.seznam.cz. Tento příkaz zobrazí trasu přes jaké servery se signál dostane od vás až na požadovanou adresu.

Pokud chcete zjistit kontakt na vlastníku určité domény, můžete využít například centrální databázi domén na adrese http://www.nic.cz, kde zadáte název domény a zobrazí se vám výpis k této doméně.

Celý Internet je rozdělen do tzv. domén několika úrovní. Jednotlivé úrovně se od sebe oddělují tečkou. Jméno se zkoumá zprava doleva. Nejvyšší doménové jméno je tzv. root doména, která se vyjadřuje tečkou zcela vpravo. Tato tečka se zpravidla vypouští a v označení se neuvádí. Doména 1. úrovně zpravidla identifikuje stát nebo generickou doménu, které se převážně používají v USA (edu, net, org apod.)

Příklad adresy: soscb.cz

cz - doména 1. úrovně – označuje stát (Česká republika)
soscb - doména 2. úrovně – označuje firmu (Střední odborná škola veterinární, mechanizační a zahradnická a Jazyková škola s právem SJZ)
Některé domény 1. úrovně:

<table>
<thead>
<tr>
<th>Austrálie</th>
<th>au</th>
<th>Belgie</th>
<th>be</th>
<th>Bulharsko</th>
<th>bg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kanada</td>
<td>ca</td>
<td>Čína</td>
<td>cn</td>
<td>Česká republika</td>
<td>cz</td>
</tr>
<tr>
<td>Německo</td>
<td>de</td>
<td>Francie</td>
<td>fr</td>
<td>Polsko</td>
<td>pl</td>
</tr>
<tr>
<td>Rusko</td>
<td>ru</td>
<td>Slovenská republika</td>
<td>sk</td>
<td>Španělsko</td>
<td>es</td>
</tr>
<tr>
<td>Taiwan</td>
<td>tw</td>
<td>Japonsko</td>
<td>jp</td>
<td>USA</td>
<td>us</td>
</tr>
<tr>
<td>Holandsko</td>
<td>nl</td>
<td>USA – vzdělávací</td>
<td>edu</td>
<td>USA – komerční</td>
<td>com</td>
</tr>
<tr>
<td>USA – vládní</td>
<td>gov</td>
<td>USA – vojenské</td>
<td>mil</td>
<td>USA – jiné</td>
<td>org</td>
</tr>
</tbody>
</table>

**URL – adresa zdroje**

URL (Uniform Resource Locator) adresa označuje v počítači přesné místo, kde se nachází požadovaný zdroj. Tento typ adres zejména využívá služba WWW. Adresa obsahuje protokol, DNS adresu počítače a popřípadě cestu v adresářové struktuře.

Příklad URL adresy:

```plaintext
ftp://pub.vse.cz/pub/netscape/navigator/3.01/windows/n16e301.exe
```

- **ftp** - název protokolu
- **pub.vse.cz** - DNS adresa počítače
- **pub/netscape.../windows** - adresářová cesta
- **n16e301.exe** - název požadovaného souboru
4 JAZYK HTML A XHTML


V další části se budeme zabívat značkovacím jazykem XHTML.

- XHTML je zkratkou EXtensible HyperText Markup Language
- XHTML je vyvinut coby nástupce HTML
- XHTML je téměř shodný se značkovacím jazykem HTML 4.01
- XHTML je přísnější a čistější než jazyk HTML
- XHTML je HTML definované jako aplikace XML
Podmínky pro používání značkovacího jazyka XHTML

- Všechny značky musí být vzájemně správně vnořeny
  `<i><b>špatně</b></i>`
  `<i><b>správně</b></i>`

- Všechny značky musí být ukončené koncovou značkou
  `<p>špatně`
  `</p>`
  `<p>správně</p>`

- U značek, u kterých není uzavírací značka, musíme danou značku uzavřít nejlépe způsobem
  `<začetka />`
  např.: `<area /> <base /> <basefont /> <br /> <col /> <frame /> <hr />`
  `<img /> <input /> <isindex /> <link /> <meta /> <param />`

- Všechny značka musíme psát malými písmeny
  `<A HREF="pokus.html">špatně</A>`
  `<a href="pokus.html">správně</a>`

- Hodnoty atributů musí být v uvozovkách
  `<td dcolspan=2>špatně</td>`
  `<td colspan="2">správně</td>`

- Používat externí skripty a šablony stylů
  `<script language="Javascrip" src="pokus.js" type="text/javascript"></script>`
  `<link rel="StyleSheet" type="text/css" href="index.css" />`
Přísně oddělujte obsah stránek od formátování.

**DTD**

DTD (**D**ocument **T**ype **D**efinition) je jinými slovy návod pro prohlížeč zpracovávající dokument. Říká mu, jaké elementy dokument používá a jak s nimi zacházet.

Jazyk XHTML má tři definice DTD, korespondující s definicemi pro HTML 4.01:

- **Strict** (striktní)
- **Transitional** (přechodnou)
- **Frameset** (s podporou frames - rámců)

Každá z těchto definic definuje jinou sadu XHTML tagů. Element DOCTYPE není součástí XHTML dokumentu, není elementem XHTML a nemusí mít tedy koncovou značku.

**Strict**

Tato definice obsahuje pouze plně podporované značky (události, atributy...) a jsou z ní vypuštěné všechny ty nedoporučované. Přicházíme tak o řadu prvků pro formátování textu. Toto formátování je možné nahradit pomocí kaskádových stylů (CSS) Cascading Style Sheets.

Striktní definice se deklaruje takto:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

**INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ**
**Transitional**

Toto je definice nejméně bolestivá při přechodu do standardu XML. Definuje naprostou většinu elementů HTML 4.01, tedy i ty nedoporučované. Její výhodou je možnost uplatnit v maximální možné míře znalosti jazyka HTML, ale hlavně dostatečná podpora současných prohlížečů.

Přechodná definice se deklaruje takto:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

**Frameset**

Tuto definici je nutné použít, pokud pracujeme s frames (s rámečky). Ostatní elementy jsou téměř totožné jako v definici Transitional.

Definice s podporou rámů se deklaruje takto:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```
5 KASKÁDOVÉ STYLY - CSS

Kaskádové styly (CSS – Cascading Style Sheets) jsou prostředkem, který zajišťuje jednotný vzhled publikovaných stránek. Technologii CSS podporují prohlížeče od verze IE 4.0.


Nástin možností CSS

- Nastavení libovolné a přesné velikost písma, prokládání, kapitálek;
- Odsazení prvního řádku odstavce, zvětšení řádkování;
- Zrušení nebo zvětšení prázdného prostoru po odstavci;
- Automatické formátování nadpisů;
- Zvýrazňování odkazů po přejetí myší;
- Automaticky grafické odrážky;
- Zneviditelnění, zprůhlednění nebo nezobrazení určité části textu;
- Předefinování grafického významu běžných tagů;
- Nastavení pozadí čehokoliv;
- Umístění libovolného objektu kamkoliv do stránky;
- Přidání rolovacích lišt;
- Orámování a nastavení okrajů;
- Dynamické prvky na stránce.

CSS styl můžete definovat třemi způsoby:

1) **in-line** zápis (přímý zápis) – slouží k nastavení parametrů konkrétní značky. Za požadovanou značkou se použije parametr STYLE=”. Hodnotou tohoto parametru je seznam atributů, které tuto značku ovlivní.

```html
<p style="color: red">Tento odstavec bude červený.</p>
```
2) **stylopis** – zapisuje se v hlavičce HTML stránky párovou značkou <style>. Uvnitř této značky se definuje značka, kterou chcete nastavit a ve složených závorkách za touto značkou je seznam atributů. Jednotlivé atributy se oddělují středníkem. Tento stylopis se používá pro nastavení jedné WWW stránky.

```
<style>
  p { color: red; }
  h1 { font-size: 20pt; color: blue; }
</style>
```

3) **externí stylopis** – používá se pro více WWW stránek. Na těchto stránkách je zaručen stejný vzhled. Tento stylopis je tvořen externím souborem (např. styl.css), vněmž je styl definován. WWW stránka musí obsahovat link na tento soubor. Pro přehlednost a budoucí úpravy vytvořených stránek se doporučuje používat externí stylopis. Pro více stránek může být jeden stylopis (Stránka vždy musí obsahovat příslušný link) nebo každá stránka může mít stylopis v jiném souboru.

Soubor **styl.css**

```
p { color: red; }
  h1 { font-size: 20pt; color: blue; }
```

WWW stránka musí obsahovat link na jméno souboru stylopisu:

```
<link rel="stylesheet" href="styl.css" type="text/css" />
```
Pro větší přehlednost stylopisu je vhodné používat komentáře. Tyto komentáře jsou zapsány mezi oddělovače /* komentář */. Parametry a hodnoty jsou psány písmeny malými.

Pokud potřebujeme nastavit společnou vlastnost pro více značek (selektorů) najednou, musí se značky od sebe oddělovat čárkou.

\[ h1, h2, h3, h4 \{\ color: red; \} \]

Je nutné všimnout si, kde se používají uvozovky, dvojtečky, složené závorky, středníky a čárky. Pokud si některé znaménko poplete, nebude to pravděpodobně fungovat. Příklad správného zápisu:

\[ h2 \{\ color: green; font-style: italic; \} \]

- h2 je selektor = jméno tagu, jehož formátování se mění
- {} složené závorky vymezují deklaraci formátu onoho selektoru
- color je vlastnost a blue jeho hodnota (barva: modrá), vlastnost a hodnota jsou odděleny dvojtečkou a mezerou
- font-style je další vlastnost a italic je její hodnota (řez-fontu: kurzíva)
- dvě deklarace se oddělují středníkem.
5.1 TŘÍDY

Třídy umožňují ve stylopizech vytvářet několik různých stylů pro jednu značku. Tyto definované styly se liší jménem své třídy. Tyto třídy jsou tzv. regulární, které mohou být použity například pro různé druhy odstavců.

```css
p.velke { text-align: left; font-size: 30pt; color: blue; }
p.male { text-align: center; font-size: 10pt; color: red; }
```

Ve vlastním kódu WWW stránky použijeme parametr class="název třídy", který uvedeme u značky.

```html
<p class="velke">
Kromě regulárních tříd lze také definovat třídy generické, které lze použít u libovolné značky.

```css
cervena { color: red; }
```

Tuto generickou třídu používáme stejným způsobem jako třídu regulární.

```html
<h1 class="cervena">
U kaskádových stylů existují ještě tzv. pseudotřídy, které se používají k definici stylu zobrazení pro určité stavy značek. Jméno značky se odděluje od názvu pseudotřídy dvojtečkou. Jednotlivé názvy tříd jsou předdefinovány, takže nemůžeme vytvářet další pseudotřídy.
Nejpoužívanější pseudotřídy u odkazů a odstavců:

- **a:link** - pro nenavštívený odkaz
- **a:active** - pro odkaz, který je právě vyvoláván
- **a:visited** - pro navštívený odkaz
- **a:hover** - pro odkaz, nad nímž se právě nachází kurzor myši
- **p:first-line** - pro zvýraznění prvního řádku odstavce
- **p:first-letter** - pro zvýraznění prvního znaku odstavce

**Příklad externího stylopisu:**

1. soubor styl.css

```css
body { background-color: yellow; } /* nastavení barvy pozadí na žlutou */

h1 { color: #000000; text-align: left; } /* stylopis, barva textu černá, zarovnávání vlevo */
font-size: 25px; /* velikost písma */
font-style: italic; /* kurzíva */

h4 { color: red; text-align: center; } /* stylopis, barva textu červená, zarovnávání na střed */
font-size: 14px; /* velikost písma */
font-style: italic; /* kurzíva */
```

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ
2. soubor – vlastní WWW stránka

```xml
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs" lang="cs">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<meta name="description" content="... obsah dokumentu..." />
<meta name="Author" content="vaše jméno" />
<title>Layout stránky</title>
<link rel="stylesheet" href="styl.css" type="text/css" />
</head>
<body>
<h1>Obor: Mechanizace a služby</h1>
<h4>Zaměření: Zemědělská mechanizace a služby</h4>
<h1>Obor: Veterinářství</h1>
<h1>Obor: Zahradnictví</h1>
</body>
</html>
```

Obr. 5

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ
Délkové jednotky používané v CSS stylech:

<table>
<thead>
<tr>
<th>Absolutní jednotky</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>px</td>
<td>Pixel (1 pixel = 1 bod obrazovky)</td>
</tr>
<tr>
<td>in</td>
<td>Palec (1 palec = 2,54cm = 72pt)</td>
</tr>
<tr>
<td>pt</td>
<td>Bod (1 bod = 1/72in = 1/12pc)</td>
</tr>
<tr>
<td>pc</td>
<td>Pica (1 pica = 12pt)</td>
</tr>
<tr>
<td>cm</td>
<td>Centimetry</td>
</tr>
<tr>
<td>mm</td>
<td>Milimetry</td>
</tr>
</tbody>
</table>

Relativní jednotky

| em                 | šířka velkého písmene M                   |
| ex                 | výška malého písmene x                    |

Jednotka, která je tvořena dvojicí písmen musí být připojena ihned za číslem (bez mezery).

Jednotky em a ex mění svou velikost v závislosti na aktuální velikosti písma. Aktuální velikost písma může být ovlivněna jednak nadřazeným nastavením stylu (font-size), formátovacími značkami (<big> a pod.) a také volbou uživatele.
Barvy používané CSS stylech:

Při zápisu barvy můžete pužívat několik způsobů. První možností je použití anglického názvu barvy a druhou možností je zápis barvy ve formátu RGB (červená, zelená, modrá). Pokud zadáváte barvu číslem musíme před číslo vložit znak #.

<table>
<thead>
<tr>
<th>Zápis</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>#rrggb</td>
<td>pro každou barvu číslo 1-16 hexadecimálně</td>
</tr>
<tr>
<td>#rgb</td>
<td>pro každou barvu číslo 1-16 hexadecimálně</td>
</tr>
<tr>
<td>rgb(r, g, b)</td>
<td>r, g, b jsou od 0 do 255</td>
</tr>
<tr>
<td>rgb(r%, g%, b%)</td>
<td>r, g, b jsou od 0 do 100</td>
</tr>
</tbody>
</table>

Například zápisy `body { background-color: blue; }` a `body { background-color: #0000ff; }` jsou shodné. V zápisu 0000FF značí první dvě hexadecimální čísla barvu červenou (R), druhá dvě čísla barvu zelenou (G) a třetí dvě čísla barvu modrou (B). Při používání barev na internetových stránkách se řídte barevnými modely. Stránky musí být kontrastní a dobře čitelné.

Příklady barev základních 16 odstínů (VGA):

- jasná bílá       white       FFFFFF
- jasná žlutá      yellow      FFFF00
- jasná fialová    fuschia     FF00FF
- jasná červená    red         FF0000
jasná tyrkysová  aqua         00FFFF  
jasná zelená    lime         00FF00  
jasná modrá     blue         0000FF  
šedá           gray         808080  
stříbrná        silver       COCOCO  
olivová        olive        808000  
purpurová      purple       800080  
základní červená maroon      800000  
základní tyrkysová teal        008080  
základní zelená  green       008000  
námořnická modř navy        000080  
černá          black       000000  

5.2 **Nejpoužívanější atributy CSS stylů**

**Vlastnosti písma**

<table>
<thead>
<tr>
<th>Popis</th>
<th>Atribut</th>
<th>Možné hodnoty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rodina písma</td>
<td><em>font-family</em></td>
<td>seznam dostupných písem</td>
</tr>
<tr>
<td>Styl písma</td>
<td><em>font-style</em></td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>italic</td>
</tr>
<tr>
<td>Obor</td>
<td>Vlastnost</td>
<td>možnosti</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Varianta písma</td>
<td>font-variant:</td>
<td>normal, small-caps</td>
</tr>
<tr>
<td>Tučnost písma</td>
<td>font-weight:</td>
<td>normal, bold, bolder, lighter</td>
</tr>
<tr>
<td>Velikost písma</td>
<td>font-size:</td>
<td>xx-small, x-small, small, medium, large, x-large, xx-large, larger, smaller, „délka“, „%“</td>
</tr>
</tbody>
</table>
### Vlastnosti pro nastavení barev

<table>
<thead>
<tr>
<th>Popis</th>
<th>Atribut</th>
<th>Možné hodnoty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Barva popředí</td>
<td>color:</td>
<td>„barva“</td>
</tr>
<tr>
<td>Barva pozadí</td>
<td>background-color:</td>
<td>„barva“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>transparent</td>
</tr>
<tr>
<td>Obrázek na pozadí</td>
<td>background-image:</td>
<td>„URL“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>none</td>
</tr>
<tr>
<td>Směr opakování obrázku na pozadí</td>
<td>background-repeat:</td>
<td>repeat</td>
</tr>
<tr>
<td></td>
<td></td>
<td>repeat-x</td>
</tr>
<tr>
<td></td>
<td></td>
<td>repeat-y</td>
</tr>
<tr>
<td></td>
<td></td>
<td>no-repeat</td>
</tr>
<tr>
<td>Pohyb obrázku (se stránkou, nebo fixovaný)</td>
<td>background-attachment:</td>
<td>scroll</td>
</tr>
<tr>
<td></td>
<td></td>
<td>fixed</td>
</tr>
<tr>
<td>Poloha obrázku na pozadí</td>
<td>background-position:</td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>top</td>
</tr>
<tr>
<td></td>
<td></td>
<td>center</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bottom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right</td>
</tr>
</tbody>
</table>
### Vlastnosti textu

<table>
<thead>
<tr>
<th>Popis</th>
<th>Atribut</th>
<th>Možné hodnoty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Velikost mezer mezi slovy</td>
<td>word-spacing:</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„délka“</td>
</tr>
<tr>
<td>Velikost mezer mezi písmeny</td>
<td>letter-spacing:</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„délka“</td>
</tr>
<tr>
<td>Čáry a blinkání</td>
<td>text-decoration:</td>
<td>none</td>
</tr>
<tr>
<td></td>
<td></td>
<td>underline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>overline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>line-trough</td>
</tr>
<tr>
<td></td>
<td></td>
<td>blink</td>
</tr>
<tr>
<td>Vertikální zarovnávání</td>
<td>vertical-align:</td>
<td>baseline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>sub</td>
</tr>
<tr>
<td></td>
<td></td>
<td>super</td>
</tr>
<tr>
<td></td>
<td></td>
<td>top</td>
</tr>
<tr>
<td></td>
<td></td>
<td>text-top</td>
</tr>
<tr>
<td></td>
<td></td>
<td>middle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>bottom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>text-bottom</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td>Kapitálky, malá a velká písmena</td>
<td>text-transform:</td>
<td>capitalize</td>
</tr>
<tr>
<td></td>
<td></td>
<td>uppercase</td>
</tr>
<tr>
<td>Vlastnosti boxů</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Popis</th>
<th>Atribut</th>
<th>Možné hodnoty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Velikost horního okraje</td>
<td>margin-top:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Velikost pravého okraje</td>
<td>margin-right:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td>Název</td>
<td>CSS značka</td>
<td>Druhy</td>
</tr>
<tr>
<td>-----------------------</td>
<td>-----------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>Velikost spodního okraje</td>
<td>margin-bottom:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Velikost levého okraje</td>
<td>margin-left:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Velikost vnitřního horního okraje</td>
<td>padding-top:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td>Velikost vnitřního pravého okraje</td>
<td>padding-right:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td>Velikost vnitřního spodního okraje</td>
<td>padding-bottom:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td>Velikost vnitřního levého okraje</td>
<td>padding-left:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td>Šířka horní části rámečku</td>
<td>border-top-width:</td>
<td>thin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>medium</td>
</tr>
<tr>
<td></td>
<td></td>
<td>thick</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„délka“</td>
</tr>
<tr>
<td>Šířka pravé části rámečku</td>
<td>border-right-width:</td>
<td>thin</td>
</tr>
<tr>
<td></td>
<td></td>
<td>medium</td>
</tr>
<tr>
<td>Nastavení vlastností horní části rámečku</td>
<td>border-top:</td>
<td>border-top-width</td>
</tr>
<tr>
<td>Šířka spodní části rámečku</td>
<td>border-bottom-width:</td>
<td>thin</td>
</tr>
<tr>
<td>Šířka levé části rámečku</td>
<td>border-left-width:</td>
<td>thin</td>
</tr>
<tr>
<td>Barva rámečku</td>
<td>border-color:</td>
<td>„barva“</td>
</tr>
<tr>
<td>Nastavení vlastností</td>
<td>pravé části rámečku</td>
<td>border-right:</td>
</tr>
<tr>
<td>----------------------</td>
<td>----------------------</td>
<td>--------------</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nastavení vlastností</td>
<td>spodní části rámečku</td>
<td>border-bottom:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nastavení vlastností</td>
<td>levé části rámečku</td>
<td>border-left:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Šířka</td>
<td>width:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Výška</td>
<td>height:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Plovoucí objekt</td>
<td>float:</td>
<td>left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right</td>
</tr>
<tr>
<td></td>
<td></td>
<td>none</td>
</tr>
<tr>
<td>Skončení plovoucího</td>
<td>clear:</td>
<td>none</td>
</tr>
<tr>
<td>objektu</td>
<td></td>
<td>left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>right</td>
</tr>
<tr>
<td></td>
<td></td>
<td>both</td>
</tr>
</tbody>
</table>
Blokový model v CSS:

![Blokový model v CSS](image)

Obr. 6

Klasifikační vlastnosti - odrážky

<table>
<thead>
<tr>
<th>Popis</th>
<th>Atribut</th>
<th>Možné hodnoty</th>
</tr>
</thead>
<tbody>
<tr>
<td>Druh elementu</td>
<td>display:</td>
<td>block</td>
</tr>
<tr>
<td></td>
<td></td>
<td>inline</td>
</tr>
<tr>
<td></td>
<td></td>
<td>list-item</td>
</tr>
<tr>
<td></td>
<td></td>
<td>none</td>
</tr>
<tr>
<td>Práce s mezerami</td>
<td>white-space:</td>
<td>normal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>pre</td>
</tr>
<tr>
<td></td>
<td></td>
<td>nowrap</td>
</tr>
<tr>
<td>Druh odrážek v seznamech</td>
<td>list-style-type:</td>
<td>disc</td>
</tr>
<tr>
<td></td>
<td></td>
<td>circle</td>
</tr>
<tr>
<td></td>
<td></td>
<td>square</td>
</tr>
<tr>
<td></td>
<td></td>
<td>decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>decimal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>lower-roman</td>
</tr>
<tr>
<td></td>
<td></td>
<td>upper-roman</td>
</tr>
<tr>
<td>Popis</td>
<td>Atribut</td>
<td>Možné hodnoty</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Způsob pozicování elementu</td>
<td>position:</td>
<td>absolute</td>
</tr>
<tr>
<td></td>
<td></td>
<td>relative</td>
</tr>
<tr>
<td></td>
<td></td>
<td>static</td>
</tr>
<tr>
<td>Posunutí elementu vpravo</td>
<td>left:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Posunutí elementu dolů</td>
<td>top:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>auto</td>
</tr>
<tr>
<td>Šířka elementu</td>
<td>width:</td>
<td>„délka“</td>
</tr>
<tr>
<td></td>
<td></td>
<td>„%“</td>
</tr>
</tbody>
</table>

Řízení pozice
<table>
<thead>
<tr>
<th>Výška elementu</th>
<th>height:</th>
<th>„délka“&lt;br&gt;„%“&lt;br&gt;auto</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definice obdelníkové části, která bude viditelná</td>
<td>clip:</td>
<td>auto&lt;br&gt;rect</td>
</tr>
<tr>
<td>Způsob zobrazení elementu</td>
<td>overflow:</td>
<td>none&lt;br&gt;clip&lt;br&gt;scroll</td>
</tr>
<tr>
<td>Pozice elementu na pseudoose Z</td>
<td>z-index:</td>
<td>auto&lt;br&gt;„číslo“</td>
</tr>
<tr>
<td>Viditelnost elementu</td>
<td>visibility:</td>
<td>inherit&lt;br&gt;visible&lt;br&gt;hidden</td>
</tr>
</tbody>
</table>
5 JavaScript

JavaScript je programovací jazyk, který se používá v internetových stránkách. Zapisuje se přímo do HTML kódu (podobně jako PHP), což je velká výhoda, protože je to jednoduché.

JavaScript je klientský skript, tzn. že k interpretaci dochází v prohlížeči (na straně klienta).

PHP je serverový skript, tzn. že k interpretaci dochází na straně serveru.

JavaScrip se někdy zaměňuje s JAVOU. Jsou to dva odlišné programovací jazyky, které mají pouze podobnou syntaxi.

Tvorba webové aplikace se může skládat z následujících částí:

- Obsahu stránky – obsah je tvořen HTML kódem
- Vzhled stránky (formátování) - formátování stránky se provádí pomocí CSS stylů
- Chování stránky – využívá se Javascript

5.1 Charakteristiky jazyka JavaScript

Jazyk JavaScript je v dnešních internetových stránkách velice rozšířený a populární. Je to jazk:

- Interpretovaný – nemusí se kompilovat, stačí interpreter
- Objektový – využívá objektů prohlížeče a zabudovaných objektů
- Závislý na prohlížeči – funguje ve většině prohlížečů. Nevýhodou může být, že zpracování stránky je mnohdy odlišné v různých typech prohlížečů
- Case sensitivní – záleží na velikosti písem v zápisu
- Syntaxi podobný jazykům C, PHP, Java a podobným.
Nevýhody jazyka

- JavaScript funguje pouze v prohlížeči
- Uživatel může JavaScript zakázat
- Existují různé odlišné verze jazyka i prohlížečů, což vede k častým chybám
- Sám o sobě neumí přístupovat k souborům (kromě cookies) ani k žádným systémovým objektům
- Sám o sobě neumí žádná data uložit (kromě cookies)
- Interpretovaný jazyk
  - Je překládán až za běhu programu
  - Je pomalejší, ale nemá tak velké formální požadavky
    Překládají se interpretrem, ten instrukce zároveň při překladu provádí a to vše na straně serveru.
- Je tzv. case sensitivní. To znamená, že záleží na velikosti písma v zápisu kódu.

5.2 **NEJČASTĚJŠÍ APLIKACE PSANÉ V JAVASCRIPTU**

- Vstupní kontrola formulářových dat
- Reakce na události vyvolané uživatelem
- Dynamické stránky
- Hodiny
- Dynamická změna obrázků
- apod.
5.3 **Schéma komunikace PHP**

![Diagram přenosu dat HTML a skriptu mezi prohlížeč a server]

Obr. 7

5.4 **Způsob zápisu JavaScriptu do dokumentu**

JavaScript se do webové stránky dá zapsat třemi způsoby:

- Pomocí značek `<script></script>` do proudu dokumentu
- Pomocí značek `<script></script>` s odkazem na externí soubor
- In-line zápisem jako atribut značky
- Kombinací výše popsaných způsobů
Při zápisu kódu je potřeba dát pozor na uvozovky. Pokud by nastala situace, že v uvozovkách budou další uvozovky, musí se vnitřní uvozovky nahradit apostrofem – " ' ... ' ".

5.5 ZÁPIS DO PROUDU DOKUMENTU

- Skript se vloží do značek

```
<script>

<body>

<h1>Zápis JavaScriptu</h1>
<p>Toto je normální odstavcový text</p>
<script>
    document.write("Toto je text zapsaný JavaScriptem");
</script>

</body>

</script>
```

Obr. 8

![Obr. 9](image)

**Zápis JavaScriptu**

Toto je normální odstavcový text
Toto je text zapsaný JavaScriptem

**INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ**
5.6 Externí zápis JavaScriptu

Je potřeba vytvořit externí soubor, který musí mít příponu .js. Tento soubor potřeba uložit do stejného adresáře jako je vlastní internetová stránka. V tomto případě lze volat externí soubor bez uvedení cesty.

- Do externího souboru s příponou .js vložte kód JavaScriptu.

Externí soubor: prvni_script.js

```javascript
1 document.write("Toto je text zapsaný JavaScriptem");
```

Obr. 10

Stránka

```html
13 <h1>Zápis JavaScriptu</h1>
14 <p>Toto je normální odstavcový text</p>
15 <script src="prvni_script.js"></script>
```

Obr. 11
5.7 **IN-LINE ZÁPIS JAVASCRIPTU**

Tento zápis se provádí přímo k dané značce

Stránka

```
13  <h1>Zápis JavaScriptu</h1>
14  <a href="http://www.soscb.cz">
15  onmouseover="alert('Toto je naše škola')">SOŠ VMZ</a>
```

Po najetí myši na odkaz se zobrazí hláška

![Zápis JavaScriptu](Obr. 12)

Obr. 13

---

**INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ**
5.8 KOMBINOVANÝ ZÁPIS JAVA SCRIPTU

Při kombinovaném zápisu je potřeba dát pozor na začátek a konec zápisu JavaScriptu. 

Toto je špatně:

```html
<script type="text/javascript" src="externi_skript.js">
    alert("funguju")
</script>
```

V tomto případě se interní skript nespustí.
Je třeba použít dva tagy

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

<script>
    alert("funguju")
</script>

Příklad – vložení menu pomocí JavaScriptu

Stránka

![Obr. 16](image)

Externí soubor – menu.js

```javascript
1 document.write("<a href='prvni.htm'>Prvni stranaka</a><br />")
2 document.write("<a href='druha.htm'>Druha stranaka</a>")
```
5.9 PŘÍKAZY JAZYKA JAVASCRIPTU

Příkazy jazyka JavaScript se oddělují:

**Koncem řádku**

První příkaz

Další příkaz

**Středníkem**

První příkaz; Další příkaz;
Komentáře

// jednořádkový
/* více Řádkový */

Komentáře nelze do sebe vnořovat
Toto je špatně
/* více Řádkový */

5.10 PROMĚNNÉ, DATOVÉ TYPY, OPERÁTORY

5.10.1 PROMĚNNÉ JAVASCRIPTU

Každá proměnná má název a hodnotu.
Operátor „=“ se používá pro přiřazení hodnoty.
Proměnné jsou case senzitive. To znamená, že je rozdíl mezi názvem proměnné „skola“ a „Skola“.

Klíčové slovo var se využívá pro deklaraci pouze globální proměnné, kterou nelze přepsat ve funkci. Logika lokální a globální proměnné je v Javascriptu odlišná, oproti PHP.

Příklady:
var prom = 23;
var jmeno="Novák";
var c, b, d; /* var se také používá, když chcete deklarovat
názvy více proměnných */
sude_cislo = 4;
mesto = "Paříž";
adresa = "Dobrého 15"; nadpis6 = "<h1>nadpis</h1>";

Proměnná se vypíše např. příkazem:
document.write(x) nebo document.write(jmeno)

5.10.2 Datové typy JavaScriptu

Textové řetězce

- 'text'
- "text"

V textových řetězcích lze používat tzv. escape sekvence - \, \, ", \n, \t

Čísla

- celá
- desetinná
- NaN (hodnota, která není číslo Not a Number)
- Infinity (nekonečno)

Pravdivostní hodnoty

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ
- 48 -

- true
- false

Jako false se vyhodnotí také null, undefined, "", 0, NaN.

**Zvláštní hodnoty**

- null (prázdná hodnota)
- undefined (nedefinovaná hodnota)

**Pole**

- [ prvek, ... ]

Pole je ve skutečnosti druh objektu.

**Objekty**

- { vlastnost: hodnota, ... }

**Funkce**

- function () { }

**Regulární výrazy**

- přepínače
5.10.3 Operátory JAVASCRIPTU

Zřetězení
  • +

Převod na číslo
  • +

Matematické
  • +
  • -
  • *
  • /
  • %
  • ++
  • --

Přiřazení
  • =
  • +=
  • -=
  • *=
  • /=

Porovnání hodnoty
  • ==
  • !=
  • <
  • <=
  • >
  • >=
Porovnání hodnoty a typu

- `===`
- `!==`

Logické

- `&&`
- `||`
- `!`
- `(podmínka ? splněno : ne)`

Bitové

- `&`
- `|`
- `^`
- `~`
- `<<`
- `>>`
- `>>>`

5.11 Hlášky v JavaScriptu

Hlášky v JavaScriptu jsou trojího typu:

- Alert
- Prompt
- Confirm
5.11.1 ALERT

Zobrazí upozorňovací okno s textem.

```
13 <script>
14 alert("Baf, baf!");
15 </script>
```

Obr. 19

![Obr. 19 Alert](image)

5.11.2 PROMPT

Zobrazí dialogové okno a čeká na zadání dat. Po zadání dat skript pokračuje v činnosti.

```
13 <script>
14 x=prompt("Zadejte jméno", "");
15 document.write("Vaše jméno je ");
16 document.write(x);
17 </script>
```

Obr. 20

![Obr. 20 Prompt](image)
5.11.3 **CONFIRM**

Zobrazí potvrzující dialog ANO/NE. Návratová hodnota je PRAVDA/NEPRAVDA – true/false.

```html
13  <script>
14    souhlas=confirm("Souhlasíte?");
15    if(souhlas) document.write("To je dobře, že souhlasíte"), exit;
16    document.write("Škoda, že nesouhlasíte")
17  </script>
```
5.12 ŘÍDÍCÍ STRUKTURY V JACASCRIPITU

Pro větvení programu se používají funkce:

- if (podmínka)
- while (podmínka)
- for (inicializace; podmínka; krok)
- switch (case; break; default)

5.12.1 IF

Syntaxe:

```javascript
if(podmínka) {
příkazy prováděné při splnění podmínky;
}
else {
příkazy prováděné při nesplnění podmínky;
}
```

Jako podmínka se musí uvést výraz, jehož logická hodnota je true nebo false (pravda nebo nepravda). Pokud podmínka nebude vracet logické hodnoty, dojde k automatickému přetypování.
Příklad na podmínku (testování prospěchu):

```javascript
var prospech="prospěl";
if (prospech=="prospěl s vyznamenáním"){
    document.write("Je to super");
}
else {
    document.write("Nic moc");
}
```

### 5.12.1.1 Ternární operátor

Ternální operátor umožňuje rychlejší zápis rozhodování, pokud chcete pouze přiřadit hodnotu proměnné.

Syntaxe:

```javascript
proměnná = podmínka ? hodnota1 : hodnota2;
```

Pokud je podmínka pravdivá, má proměnná hodnotu hodnota1, pokud je podmínka nepravdivá, má proměnná hodnotu hodnota2.

```javascript
Vyber = prospech = "prospěl s vyznamenáním"? "Je to super" : "Nic moc";
```
### 5.12.2 While

Cyklus s podmínkou na začátku. Sekvence vnitřních příkazů se provádí dokola, dokud platí podmínka.

Jakmile podmínka neplatí, pokračuje program dál. Není-li splněná podmínka, neprovede se cyklus ani jednou.

**Syntaxe:**

```javascript
while (podmínka) {
    sekvence příkazů
}
```

**Příklad:**

Výpis čísel 1 až 10

```javascript
var i = 0;
while (i <= 9) {
    i++;
    document.write(i);
}
```

### 5.12.3 For

Příkaz for je cyklus s daným počtem opakování.

**Syntaxe:**

```javascript
for (počáteční hodnota; podmínka; navýšení) { příkazy; }
```
Příklad:

Výpis čísel 1 až 10

```javascript
var i=0
for (i=1; i<=10; i++)
document.write(i);
```

5.12.4 **SWITCH**

Příkaz switch se používá pro větvení do více alternativ.

**Syntaxe:**

```javascript
switch(proměnná) { case hodnota : příkaz; break;
case hodnota2 : příkaz 2; break;
...
default : příkaz n }
```

Příklad:

Oblíbený školní ročník

```javascript
switch (oblibenyRocnik) {
case "prvni":
document.write("Můj oblíbený ročník je první!"); break
```
case "druhy":
    document.write("Můj oblíbený ročník je druhý!"); break

case "treti":
    document.write("Můj oblíbený ročník je třetí!"); break

case "ctvrty":
    document.write("Můj oblíbený ročník je čtvrtý!"); break

default:
    document.write("Nemám žádný oblíbený ročník, nerad chodím do školy!");

\}

5.13 ŘÍZENÍ BĚHU PROGRAMU

Cykly lze přerušit pomocí break a continue.

**Break**

- Příkaz break předčasně ukončí cyklus while nebo for.

**Continue**

- Příkaz continue skočí na začátek těla cyklu, všechny příkazy za continue už se neprovedou a pokračuje se dále.
5.14 SKRIPTY REAGUJÍCÍ NA UDÁLOST

Jsou to skripty, které reagují na nějakou událost uživatele:

- Kliknutí
- Přejetí myši
- Načtení stránky
- Opuštění stránky
- Změna velikosti
- Stisknutí tlačítka
- apod.

5.14.1 UDÁLOSTI MYŠI

<table>
<thead>
<tr>
<th>Vlastnost</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>onClick</td>
<td>kliknutí na prvek</td>
</tr>
<tr>
<td>onDoubleClick</td>
<td>dvojklik na prvek</td>
</tr>
<tr>
<td>onMouseOver</td>
<td>najeti myši na prvek</td>
</tr>
<tr>
<td>onMouseOut</td>
<td>odjetí myši z prvku</td>
</tr>
<tr>
<td>onMouseMove</td>
<td>pohyb myši nad prvkem</td>
</tr>
<tr>
<td>onMouseDown</td>
<td>stisknutí tlačítka nad prvkem</td>
</tr>
<tr>
<td>onMouseUp</td>
<td>uvolnění tlačítka nad prvkem</td>
</tr>
</tbody>
</table>
Příklady:

Při přejetí myši nad prvním textem se zobrazí hláška. Při dvojkliku na druhý text se zobrazí hláška.

```html
13 <h1 onMouseOver="alert('Přeješ jsi myší')">14  Přejeď myší</a><br />
15 <h1 onMouseDown=alert('Dvakrát jsi kliknul');">16  Dvakrát klikni na text</h1>
```

Obr. 25

Při přejetí myši nad obrázkem se obrázek vymění, při opuštění myši se vrátí zpět původní obrázek.

```html
13 <img src="img0001.jpg" onmouseover="src='img0002.jpg'" 14  onmouseout="src='img0001.jpg'">
```

Obr. 26

Změna barvy odkazu po najetí myši na odkaz.

```html
13 <a href="http://www.soscb.cz" onMouseOver="style.color='red'">14  SOŠ VMZ</a>
```

Obr. 27
5.14.2 ÚDÁLOSTI STRÁNKY A OKNA

<table>
<thead>
<tr>
<th>Vlastnost</th>
<th>Popis</th>
</tr>
</thead>
<tbody>
<tr>
<td>onLoad</td>
<td>úplné načtení stránky</td>
</tr>
<tr>
<td>onUnload</td>
<td>opuštění stránky</td>
</tr>
<tr>
<td>onResize</td>
<td>změna velikosti okna</td>
</tr>
<tr>
<td>onScroll</td>
<td>posouvání okna</td>
</tr>
</tbody>
</table>

Příklady:

Zákaz změny textového pole ve formuláři.

```
13  <form>
14  <input type="text" value="Neměňte obsah pole"
15     onChange="alert('Nemáte měnit toto pole!!'),
16           value='Neměňte obsah pole'>
17  </form>
```

Obr. 28

Zamezení kopírování údajů ze vstupního pole.

```
13  <form>
14  <input type="text" value="Zákaz kopírování"
15     onSelect="alert('Chceš to okopírovat? Zapomeň!!'),
16          value='Zákaz kopírování'>
17  </form>
```

Obr. 29
5.15 Funkce

Vlastní funkce tvoří posloupnost instrukcí, které potřebujeme v programu na různých místech zopakovat.

Nejprve je potřeba funkci deklarovat:

```javascript
function nazevFunkce(parametr1, parametr2) {
    /* tělo funkce */ return true;
}
```

Poté zavolat jejím názvem:

```javascript
vysledek = nazevFunkce();
```

5.16 Pole

Pole je skupina proměnných, které se navzájem doplňují (jména žáků, známky, zameškané hodiny).

Každé pole má své jméno a jeho prvky jsou přístupné pomocí tzv. indexu. Index se zpravidla vkládá do hranatých závorek.

**Index standardně začíná od nuly.**

Deklarace pole:

```javascript
pole1 = new Array("Pavel", "Petr", "Dana")
pole2 = ["Pavel", "Petr", "Dana"]
```
Ke každému prvku můžeme přistupovat prostřednictvím jeho indexu, které jsou zpravidla číselné a jsou od nuly.

document.write(pole1[0])  // vypíše Pavel

Počet prvků v poli

Vlastností array.length lze zjistit počet prvků pole.

pole2 = ["Pavel", "Petr", "Dana"];
document.write(pole2.length);  // vypíše 3

Procházení pole

Procházet pole je možné pomocí cyklu for.

pole2 = ["Pavel", "Petr", "Dana"]
for(i=0;i<pole2.length;i++)
{
  document.write(pole2[i] +" ")
}

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ
5.16.1 **Asociativní pole**

Asociativní pole je pole, které je indexované řetězcem místo číslem.

```javascript
mesto = new Array();
mesto["C"] = "České Budějovice";
mesto["L"] = "Lišov";
mesto["P"] = "Praha";
mesto["O"] = "Ostrava";
```

Procházení asociativního pole

```javascript
for (var index in mesto) {
    document.write (mesto [index]);
}
```

5.17 **Objektový model JavaScriptu**

JavaScript je jazyk objektový, třebaže nevyužívá všechny možnosti OOP.

V praxi často znamená "objektovost" javascriptu vlastně jenom to, že všechny vlastnosti a příkazy jsou uspořádány podle nějakého systému.
Objektový model je způsob, jak pojmenovat jednotlivé prvky okna prohlížeče a dokumentu, aby se s nimi dalo pracovat.

K adresování objektů se používá tečková syntaxe objektů.

Většina objektů má:

- podobjekty
- vlastnosti
- metody

Zápis objektů

`objekt.podobjekt`

`objekt.vlastnost`

`objekt.metoda()`

Příklady:

Vlastnosti mohou být proměnné nebo jiné objekty a sadě funkcí spojených s objektem řikáme metody.

```
jmeno_objektu.jmeno_vlastnosti
```

document.bgColor = "red";

`window.location.href = "http://www.centrum.cz";`
jmeno_objektu.jmeno_metody

document.write("write je metodou objektu dokument.");
window.history.back(); (je zápis příkazu, který funguje stejně jako tlačítko zpět v prohlížeči)

Javascript umí přistoupit k těmto objektům a prvkům:

- K objektům okna prohlížeče (window)
- Přes něj i k prvkům stránky (window.document), které mají úzkou vazbu na jazyk HTML
- K zabudovaným objektům (Date, Math, String, Array, RegExp).
- K vlastním vytvořeným objektům

Vestavěné objekty

- Array
- Date
- Math
- RegExp
- String
5.18 Datum a čas

V JavaScriptu je datum a čas obsažen v objektu Date.

Výpis aktuálního data z počítače:

document.write(new Date());

Výpis datumu:

datum=new Date();
mesic=datum.getMonth()+1; //leden je 0
den=datum.getDate();
document.write("Dnešní datum je "+den+. "+mesic+."));

Zjištění části data

<table>
<thead>
<tr>
<th>Vlastnost</th>
<th>Hodnoty</th>
<th>Význam</th>
</tr>
</thead>
<tbody>
<tr>
<td>getFullYear()</td>
<td>rok</td>
<td>4 číslice</td>
</tr>
<tr>
<td>getMonth()</td>
<td>měsíc</td>
<td>leden je nula</td>
</tr>
<tr>
<td>getDate()</td>
<td>číslo dne v měsíci</td>
<td>1 je prvního</td>
</tr>
<tr>
<td>getDay()</td>
<td>číslo dne v týdnu</td>
<td>neděle je nula</td>
</tr>
<tr>
<td>getHours()</td>
<td>počet hodin od půlnoci</td>
<td>0 je půlnoc a hodina po ní</td>
</tr>
<tr>
<td>Funkce</td>
<td>Popis</td>
<td>Výsledek</td>
</tr>
<tr>
<td>-----------------</td>
<td>-----------------------------------------------------------------------</td>
<td>---------------------------</td>
</tr>
<tr>
<td>getMinutes()</td>
<td>počet minut od začátku hodiny</td>
<td>0 je první minuta, 59 poslední</td>
</tr>
<tr>
<td>getSeconds()</td>
<td>počet sekund od začátku minuty</td>
<td></td>
</tr>
<tr>
<td>getMilliseconds()</td>
<td>počet milisekund od začátku sekundy</td>
<td></td>
</tr>
<tr>
<td>getTime()</td>
<td>počet milisekund od 1. 1. 1970</td>
<td></td>
</tr>
</tbody>
</table>

6 AJAX – ZÁKLADNÍ UŽITÍ

Příklad:

```javascript
var xmlhttp;

if (window.XMLHttpRequest) {
    xmlhttp = new XMLHttpRequest();
}

xmlHttp.open('GET', 'stranka.php', false);
xmlHttp.send();
```
6.1 POŽADAVEK TYPU POST

Pokud chcete posílat na server formulářová data, musíte přidat HTTP hlavičku a serializovaný řetězec do metody send().

V případě požadavku typu POST musíme předat parametry metodě „send“ objektu XMLHttpRequest

- xmlhttp.send('...');

Argument funkce musíme serializovat dle vzoru:

- název=hodnota&nazev2=hodnota

Pokud předáváme data HTML formuláře, musíme přidat hlavičku

- xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");

PŘÍKLAD

```javascript
xmlhttp.open("POST", "zpracovani.php", true);
xmlhttp.setRequestHeader("Content-type","application/x-www-form-urlencoded");
xmlhttp.send("jmeno-Martin&prijmeni-Mike");
```

6.2 OBJEKT XMLHttpRequest

Objekt XMLHttpRequest přejímá 3 parametry.

- Typ požadavku (GET, POST)
- Adresa serveru
- A zdali je požadavek asynchronní či nikoliv (TRUE, FALSE)
Pokud je úloha na serveru časově náročná, běh aplikace se zastaví, než se daná úloha vykoná. S využitím technologie AJAX (zasláním asynchronního požadavku) k zastavení aplikace nedojde.

Příklad použití ASynchronního požadavku:

```javascript
xmlhttp.open("GET", "stranka.php", true);
```

### 6.3 Odpočívá serveru

Odpočívá serveru je uložená ve vlastnosti responseText nebo responseXml objektu XMLHttpRequest.

Pokud webový server vrací odpověď ve formátu HTML, můžeme rovnou uložit tento výsledek do HTML elementu v dokumentu.

Příklad responseText:

```javascript
document.getElementById("kontejner").innerHTML = xmlhttp.responseText;
```

Příklad responseXml:

```javascript
xmlDoc = xmlhttp.responseXML;
txt="";
x = xmlDoc.getElementsByTagName("UMELCI");
for (i = 0; i < x.length; i++) {
    txt = txt + x[i].childNodes[0].nodeValue + "<br>");
}
document.getElementById("kontejner").innerHTML = txt;
```

V případě, že skript na server vygeneruje XML musíme ho i my zpracovat jako XML vstup.
6.4 **ONREADYSTATECHANGE událost**

Když odešleme na server požadavek, často chceme provést určité akce na základě odezvy. K tomu nám slouží událost „*onreadystatechange*“ událost.

Tato javascriptová událost je vykonána vždy, když se změní status vlastnost readyState objektu XMLHttpRequest.

XMLHttpRequest objekt má tři základní důležité vlastnosti:

1) onreadystatechange
   - ukládá ukazatel na funkci, která se provede při změně statusu

2) readyState
   - udržuje status požadavku a prochází od 0 do 4
   
   0: požadavek nebyl nainicializován
   1: připojení na server založeno
   2: požadavek obržen
   3: zpracovávání požadavku
   4: požadavek dokončen a odezva je připravená k použití

3) status
   - 200, 404 (HTTP status)

Když readyState vlastnost je rovna hodnotě 4 a status je roven hodnotě 200, tak víme, že odezva serveru je připravena k použití.

Příklad

```javascript
xmlhttp.onreadystatechange = function() {
    if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
        document.getElementById("div").innerHTML = xmlhttp.responseText;
    }
}
```
7 AJAX A JAVASCRIPTOVÉ FRAMEWORKY

Vzhledem k tomu, že práce s nativními funkcemi je velice náročná doporučuje se používat nějaký javascript framework, který práci s třídami obalí do smyslnějšího používání. K tomu účelu a plno dalším funkcionalitám slouží například jQuery. Ve frameworku jQuery je práce s asynchronními požadavky skutečně jednoduchá a přehledná.

jQuery automaticky vytvoří objekt, který spravuje spojení, nainicializuje správně metodu open a onreadystatechange, která v sobě obsahuje přepínač, aby se naše metoda provedla pouze v případě úspěchu.

7.1 UKÁZKA JEDNODUCHÉHO AJAX CHATU – AUTOR MÍKA M.

HTML

```html
<!DOCTYPE html>
<html>
<head>
  <title>AJAX Chat</title>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  <link rel="stylesheet" href="styl.css">
  <script type="text/javascript" src="http://code.jquery.com/jquery-2.0.2.min.js"></script>
  <script type="text/javascript" src="javascript.js"></script>
</head>
<body>
  <center><h1>Ukázka AJAX chatu</h1></center>
```
Text zprávy:
<input type="text" name="zprava" id="zprava" />
<button id="button-odeslat">Odeslat</button>
<div id="pocitadlo">
    K obnovení dojde za <span class="cislo">5</span>...
</div>

</div>
</div>
</body>
</html>

**PHP**

```php
<?php
$nazevDatovehoSouboru = 'data.txt';
if (!file_exists($nazevDatovehoSouboru)) {
    file_put_contents($nazevDatovehoSouboru, '');
}
if (!empty($_POST['zprava'])) {
    $novyRadek = date('d.m. (H:i)') . ' : ' . $_POST['zprava'] . "\n"
    file_put_contents($nazevDatovehoSouboru, $novyRadek, FILE_APPEND);
}
echo file_get_contents($nazevDatovehoSouboru);
```

**JAVASCRIPT**

```javascript
$(document).ready(function() {
    var intervalObnoveni = 5000;
    function obnovitOkno() {
        $('#vypis').load('zpracovani.php');
    }
    function nastavitCas() {
        var $cislo = $('#pocitadlo span');
        var noveCislo = parseInt($cislo.text()) - 1;
```
if (noveCislo == -1) {
    novCislo = intervalObnoveni / 1000;
    obnovitOkno();
}
$cislo.text(noveCislo);
}

obnovitOkno();
setInterval(nastavitCas, 1000);
$('#button-odeslat').click(function() {
    obnovitOkno();
    $.post('zpracovani.php',
        'zprava=' + $('#zprava').val(),
        function(responseData) {
            $('#vypis').html(responseData);
            $('#zprava').val('');
        }
    );
});

CSS

#inner { width: 1000px; margin: 0 auto; border: 1px solid #ccc; padding: 10px; height: 500px
}
#zprava { width: 300px; margin-left: 30px }
#vypis { margin-top: 10px; padding: 30px; border-top: 1px solid #ccc }
#pocitadlo { float: right; width: 200px; line-height: 30px }
8 **SEZNAM POUŽITÉ LITERATURY**


Internetové zdroje:


Použité obrázky:
Vlastní