6 ZÁKLADNÍ TECHNOLOGIE PRO TVORBU WWW

RYCHLÝ NÁHLED KAPITOLY

V této kapitole se seznámíte s hypertextem, což je základní stavební kámen pro tvorbu www stránek. Seznámíte se rovněž s jazykem HTML pomocí kterého se www stránky vytvářejí. Protože dnešní webové stránky jsou dynamické, seznámíte se i s JavaScriptem, jako základním nástrojem pro tvorbu dynamických stránek. Naučíte se rovněž základy tvorby www stránek a seznámíte se se základními nástroji pro jejich vytváření.

CÍLE KAPITOLY

Cílem kapitoly je naučit studenty tvorbě jednoduchých www stránek a možnostem jejich publikování.

ČAS POTŘEBNÝ KE STUDIU

V této kapitole je vhodné si uváděné příkazy prakticky odzkoušet. Z tohoto důvodu je čas potřebný ke studiu v rozsahu 2-5 hodin, podle míry praktické práce.

KLÍČOVÁ SLOVA KAPITOLY

Počítačová síť, TCP/IP protokol, ISO/OSI model, maska sítě, brána, IP adresa, služby Internetu, komunikace

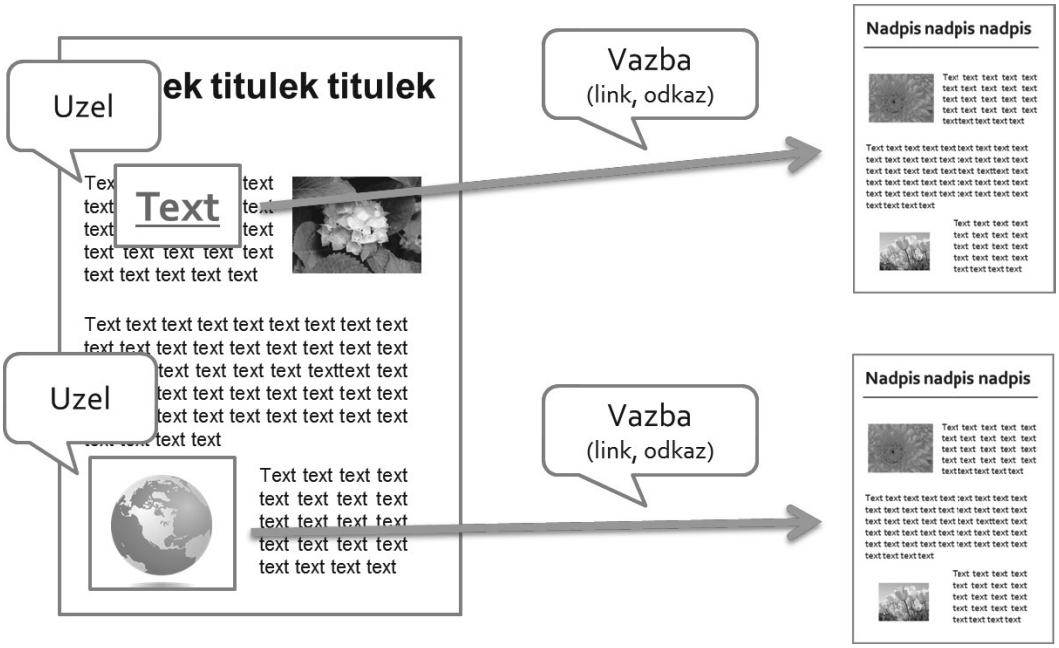
6.1 Princip Hypertextu

Hypertext je v podstatě text obsahující odkazy na další informace. Na rozdíl od lineárního textu, který známe z klasických dokumentů, umožňuje tzv. nesekvenční přístup k informacím. Zobrazení dalších informací je pak umožněno prostřednictvím odkazů.

Jak znázorňuje následující obrázek, *základní jednotkou informace* je dokument (www stránka). Informace jsou uloženy v síti uzlů propojených asociativními vazbami. Uživatel se pak může libovolně mezi nimi pohybovat pomocí vazeb.

Vazby (link, hyperlink) jsou v podstatě odkazy na další objekty. Mohou směřovat na jakékoliv místo lokálně v počítači nebo v internetu (www stránka, obrázek, multimediální soubor, nezobrazitelné soubory všech typů formátů).

Uzly jsou pak vyznačená místa v dokumentu, které reprezentují dané odkazy. Jedná se často o zvýrazněný text, netextové informace (obrázky, animace, další multimediální soubory), v HTML 5 to mohou být také blokové elementy.



Obrázek 61 Technologie hypertextu

Zdroj: Botlík, Slaninová, 2014, Služby Internetu a internetové systémy, výukový materiál OPF v Karviné

6.2 Služba WWW

Zřejmě nejpoužívanější službou internetu je služba WWW (World Wide Web). Jedná se o službu využívající internetový protokol http, popř. https. Tato služba využívá technologie hypertextu, která byla popsána v předchozí kapitole.

World Wide Web, v doslovném překladu „světová rozsáhlá síť“, „celosvětová síť“ je označení pro systém prohlížení, ukládání a odkazování dokumentů nacházejících se v Internetu. Dokumenty (webové stránky) si prohlížíme pomocí webového prohlížeče, jsou uloženy na webových serverech a jsou navzájem propojeny pomocí hypertextových odkazů

zapisovaných ve formě URL (například http://www.seznam.cz nebo http://www.google.com). Webové stránky jsou popsány pomocí HTML jazyka a pro jejich přenos mezi počítači je používán HTTP protokol.

Jak vyplývá ze samotného principu hypertextu, umožňuje tato služba integrovaný pohled na data a informace z různých zdrojů (multimediální přístup).

Jejími charakteristickými znaky jsou:

- Nezávislost na platformě, operačním systému či jediné firmě
- Pružnost
- Neustálý vývoj standardů a doporučení

6.2.1 ZÁKLADY TVORBY WEBU

Tvorba webových prezentací je obvykle v kompetenci tzv. webmastera. Webmaster zastává v rámci procesu tvorby webu několik funkcí současně:

- **Tvůrce obsahu** – vytváření, úprava HTML dokumentů, vkládání dat (internetové aplikace, údržba integrity obsahů
- **Návrhář** (webdesigner) – tvorba layoutu www stránek, tvorba grafiky, rozhoduje o celkovém vzhledu a navigační struktuře webu
- **Programátor** – tvorba www stránek pomocí samotných jazyků, ať už skriptovacích nebo jiných (PHP, ASP, CGI, Java, JavaScript, Flash, .Net)
- **Administrátor** (správce) – stará se o bezpečný provoz www serveru, o bezpečnost provozovaných skriptů, o zálohování dat, o bezpečnost důvěrných informací
- **Public relations** – člověk, který se stará o marketing a styk se zákazníkem, případně s ostatními uživateli webu

U menších webů většinou všechny tyto funkce zastává jeden člověk sám, u rozsáhlejších websitů mohou jednotlivé funkce zastávat různí lidé.

Než přistoupíme k vlastní tvorbě webu, musíme zde zmínit několik základních pojmů.

Internetová stránka

Pod pojmem internetová stránka rozumíme jakoukoliv stránku, která je součástí webu, bez ohledu na to, jakým způsobem byla vytvořena.

Web (website)

Web, popř. website je v podstatě kolekce internetových stránek, která tvoří jeden celek. Vyznačuje se určitou strukturou, danou propojením stránek pomocí odkazů. Struktura webu bývá často zobrazována v podobě mapy webu.

Domovská stránka (homepage)

Výchozí stránka, která se zobrazí jako první, zadáme-li do prohlížeče název domény bez udání názvu souboru. Většinou se jedná o soubor s názvem index.html nebo default.htm (podle operačního systému, na kterém běží web server).

6.2.2 INTERNETOVÁ STRÁNKA

Někdy také HTML stránka (podle základního jazyka používaného pro tvorbu www stránek). Jedná se o ucelený soubor informací, předávaný uživateli. Může být reprezentován ve dvou tvarech:

- **zdrojový kód –** tvar, ve kterém je stránka uložena v počítači (na webovém serveru), je psán v některém jazyce pro popis dokumentu (HTML, CSS atd.)
- **vizuální tvar –** tvar, ve kterém je stránka prezentována uživateli prostřednictvím prohlížeče.

Zobrazování internetové stránky prostřednictvím prohlížeče probíhá následujícím způsobem: nejprve dojde ke stažení a otevření souboru se zdrojovým kódem stránky (soubory s příponami .htm, .asp či .php). Dále dojde k zobrazení částí obsažených ve zdrojovém kódu (např. text). Nakonec dochází k načítání objektů, na které jsou v kódu uvedeny odkazy (např. grafika).

6.2.3 TYPY WEBOVÝCH PREZENTACÍ

Webové prezentace můžeme dělit na dvě základní skupiny: statické a dynamické. Statické webové prezentace jsou typické prezentace elektronických dokumentů. Využívají všech výhod hypertextu, nereagují však kromě zobrazení zdroje odkazu na požadavky uživatele. Naopak dynamické webové prezentace jsou typické svou uživatelskou interaktivitou, často zobrazují informace proměnlivé v čase, bývají automaticky generovány. Jako příklad můžeme uvést stránky vyhledávačů, blogy, e-shopy, webové aplikace apod.

6.3 Technologie pro vytváření webových stránek

V současné době existuje nepřeberné množství technologií, které lze použít pro tvorbu webových stránek či aplikací. Podíváme-li se na ně z hlediska tvorby statických či dynamických stránek, pro statické webové stránky je typické použití kombinace HTML, XHTML a CSS. Aktuálně je standardem HTML verze 5 a CSS3.

Dynamické webové stránky se pak liší v tom, zda jsou použité technologie zpracovávány na straně klienta nebo na straně serveru. Mezi technologie zpracovávané na straně klienta můžeme zařadit JavaScript a všechny jeho další mutace (Jscript, VBScript), ECMA Script, dále také XML, Flash či Java-applety nebo ActiveX prvky.

Na straně serveru se pak setkáme s jazyky jako je Perl či CGI skripty, skriptovací jazyky PHP, ASP, Server Side JavaSript či ASP.Net. Z dalších můžeme ještě zmínit např. Python, ColdFusion, Java Server Pages či kombinaci několika technologií, v poslední době velmi oblíbený AJAX.

6.3.1 NÁSTROJE PRO TVORBU WEBOVÝCH PREZENTACÍ

Pro tvorbu webových prezentací budeme potřebovat několik málo nástrojů. Pro samotné vytváření webových stránek postačí některý z editorů. Můžeme se setkat se dvěma základními kategoriemi editorů určených pro tvorbu webových stránek.

WYSIWYG editory (What You See Is What You Get) jsou editory, které umožňují vizuálně zobrazovat webovou stránku přímo v procesu jejího vytváření. Webmaster může vkládat na webovou stránku jednotlivé prvky pomocí sofistikovaných nástrojů, často poskytují nepřeberné množství již připravených prvků a šablon. Kdybychom šli do důsledků, při tvorbě pomocí WYSIWYG editorů není nutné znát podrobnosti o jednotlivých příkazech zvoleného jazyka. Editor totiž daný kód vytváří automaticky sám. V mnoha případech bohužel velmi nepřehlednou formou. Z čehož vyplývá, že pokud se chcete profesionálně věnovat tvorbě webových stránek, je vhodné se také dozvědět o používaných jazycích co nejvíce. Jako příklad WYSIWYG editorů můžeme zmínit FronPage Express či Adobe Dremweaver z řady komerčního softwaru, Nvu pak jako zástupce softwaru nekomerčního. WYSIWYG editorům a tvorbě www stránek jejich pomocí bude věnována samostatná kapitola.

NonWYSIWYG editory jsou ve velké míře používány profesionálními webdesignery. Hlavním důvodem, proč používat nevizuální editory, tedy editory pracující pouze s kódem, je absolutní kontrola nad napsaným kódem. Existuje mnoho editorů tohoto typu, často používaných také programátory pro svou schopnost vizuálně zobrazovat psaný kód, některé editory jsou schopny také syntaktické kontroly napsaného kódu. Po vizuální stránce je pak možné webovou stránku zobrazit přímo v prohlížeči. Jako příklad můžeme uvést PSPad Editor, WebPage Maker, HTML Kit, Ace Html, či starší HomeSite. V neposlední řadě si můžeme vystačit také s poznámkovým blokem.

Kromě editorů pro samotnou tvorbu webových stránek potřebuje tvůrce webu také software pro práci s grafikou (pro úpravu obrázků, vytváření ikon, pozadí, animací, tlačítek apod.). Zde můžeme zmínit opět několik zástupců jak komerčního softwaru (Adobe Photoshop, Adobe Illustrator, Macromedia Fireworks), tak softwaru nekomerčního (Gimp, InkSpace, IrfanView).

Pro umístění webových stránek na webovém serveru je pak vhodné mít nainstalovaného FTP klienta pro snazší a rychlejší přenos souborů z lokálního počítače na webový server. Existují sice poskytovatelé webhostingu, kteří nabízejí přímé nahrání souborů přes webové rozhraní, tato možnost se však stává v případě většího množství souborů uživatelsky velmi nepřívětivou. FTP klientů existuje nepřeberné množství, od samostatných aplikací (CuteFTP, AceFTP, WS FTP), až po FTP klienty, kteří jsou součástí jiného softwaru (Total Commander).

6.4 Práce s hypertextem

Hypertexty bývají realizovány pomocí značkovacích jazyků. Základní normou je GML (Generalized Markup Language). Jazyk SGML (Standard Generalized Markup Language), který je definován v ISO normě 8879 z roku 1986 byl příliš obecný, umožňoval definici vlastních značkovacích jazyků (sad značek a jejich vzájemných vztahů) pomocí tzv. definic typu dokumentu (DTD) a má obsáhlou množinu volitelných parametrů, počínaje maximální délkou názvů značek a konče určením znaků použitelných jako oddělovače značek od textu. Komplexnost a přílišná volnost standardu SGML zbrzdila jeho praktické využití. To přišlo s definováním HTML.

6.4.1 HTML

Každá stránka je na disku nebo na serveru uložena ve formě zdrojového kódu. Tento kód je psaný v jazyce HTML. Pomocí jazyka HTML se tvoří webové stránky. HTML znamená HyperText Markup Language - hypertextový (hypertext = odkaz) značkovací jazyk. Stránky jsou tedy zapsány pomocí zdrojového kódu. Webové stránky vytváříte pomocí HTML značek (značka, nebo-li také tag). Značek je mnoho a vždy se zapisují do špičatých závorek

HTML je nejznámější aplikací SGML (Standard Generalized Markup Language) Jedná se tedy o značkovací jazyky.

Podstata a principy značkovacích jazyků vychází z faktu, že v každém dokumentu existuje vlastní obsah a formáty, prostřednictvím kterých je obsah zobrazen. Dále existuje v dokumentech množina znaků, které jsou charakteristické pro prostředí, ve kterém byl dokument vytvořen. Například dokument napsaný ve Wordu obsahuje mimo textové informace taky údaje o znakové sadě, o parametrech stránky, o autorovi apod. tyto údaje jsou zaznamenány pomocí speciálních znaků programu Word. Pokud by dokument byl otevřen v jiném programu, tento by nevěděl, jak interpretovat řídící znaky a formáty Wordu.

Z toho plyne, že pro Internet, jako pro multiplatformní prostředí, je nutné zabezpečit, aby formátovací a řídící znaky měly jednotně interpretovatelnou podobu. Toho se dosáhlo tak, že došlo k oddělení obsahu, který je psán prostým, neformátovaným textem. Řídící a

formátovací znaky jsou rovněž zapsány prostým textem ve formě značek, které obsahují informace pro program, kterým je dokument zobrazován. Značky jsou většinou „párové“, tj. mají část, která se uvádí před textem, kterého se týká a část, která se umísťuje za text a určuje konec platnosti značky. Aby se rozlišilo, že se jedná o počáteční nebo koncovou značku, má koncová značka jiný tvar.

Například zápis *** text *** obsahuje značku , jejíž koncový tvar je a říká, že text mezi značkami se má zobrazit tučně. Množinu značek, kterou HTML používá pro označení formátů a informací příslušné značky, určuje příslušné DTD (definice typu dokumentu), které definuje verzi HTML. WWW stránky zapsaná v HTML jazyce tvoří tzv. zdrojový kód. Ten lze upravit prostřednictvím libovolného textového editoru, např, poznámkového bloku a prohlédnout v libovolném prohlížeči tak, že z menu prohlížeče vyvoláme příkaz Zobrazit > Zdrojový kód (případně View > Source, apod.).

Pro psaní HTML dokumentu platí několik zásad. Každý dokument je uzavřený (začíná a končí) do značky <html> </html>.

Dokument má dvě základní části, hlavičku <head> a tělo <body>. V hlavičce jsou obsaženy informace o dokumentu, vlastní text je v části <body>.

Dvojici značek a textu, který je ve značkách uzavřený se říká obecně element, u HTML se mu říká Tag. Tagy jsou párové (obsahují dvojici značek, počáteční a konečnou, které označují místo pro realizaci značky) a mohou se do sebe vnořovat. Obecně platí, že se tagy nesmí křížit (tag nemůže skončit, pokud neskončily všechny tagy v něm vnořené). Při psaní značek se rozlišuje velikost písma. Součástí značek jsou atributy, které dále značku upřesňují. HTML toleruje nepárovost (neuvedení konečné značky u párového tagu), křížení i nedodržení velikosti písma, obecně je to však ve značkovacích jazycích hrubá chyba. HTML definuje i množinu nepárových značek.

Základy HTML se dají shrnout následovně, dokument v jazyku HTML má předepsanou strukturu. Ta obsahuje:

Deklaraci typu dokumentu **–** značka <!DOCTYPE html> (sděluje prohlížeči, že otevřel HTML dokument).

Kořenový element **–** prvek html (značky <html> a </html>) (reprezentuje HTML dokument).

Hlavičku dokumentu **–** prvek head (značky <head> a </head>) (obsahuje metadata, která se vztahují k celému dokumentu. Definuje kódování, název dokumentu, autora, popis, klíčová slova, titulek dokumentu nebo kaskádové styly).

Tělo dokumentu **–** prvek body (značky <body> a </body>) (zahrnuje vlastní obsah dokumentu).

V konfrontaci s XML lze snadno vyvodit, že HTML je specifický XML dokument.

6.4.2 VERZE JAZYKA HTML

Verze 0.9 – 1.2 (1991–1993), nepodporují grafické rozhraní. Verze 2.0 byla vydána v listopadu 1995, odpovídá syntaxi SGML. Přidává k původní specifikaci interaktivní formuláře a podporu grafiky. Verze 3.2 (1997, komunita W3C) přidává tabulky, zarovnávání textu a stylové prvky pro ovlivňování vzhledu. Původně připravovaná verze HTML 3.0 nebyla nikdy přijata jako standard, protože byla příliš složitá. Verze 4.0 (1997, W3C). Do specifikace jazyka přibyly nové prvky pro tvorbu tabulek, formulářů a byly standardizovány rámy (frames).

V této verzi lze charakterizovat význam (sémantiku) jednotlivých částí dokumentu, a vzhled (styly). Verze byla revidovaná, verze 4.01 (1999, W3C) opravuje některé chyby, mělo se jednat o poslední verzi HTML, další specifikace měla být XHTML (verze 1.0, 1.1, 2.0 resp. 5.0), následník HTML, využívající univerzální jazyk XML. HTML 5.0 ukončuje závislost HTML na SGML opravuje chyby předešlé verze, vyřazuje zastaralé prvky, přidává nové sémantické prvky, přidává podporu nových a moderních technologií, zavádí nový systém vývoje jazyka (nové verze vydávány zhruba ve dvouletých cyklech).

HTML je samo o sobě pouze značkovací jazyk k vytváření struktury dokumentů. Ovšem s nástupem jeho poslední verze se zažitě používá označení HTML5 pro celý balíček, do kterého spadají jak kaskádové styly CSS, což je samostatný jazyk na stylování dokumentů, tak JavaScript (správný název je ECMAScript – označení JS je pouze lidové), který je taktéž samostatným skriptovacím jazykem na straně uživatele. Ve skutečnosti, bychom měli správně označovat novou verzi technologií jako HTML5, CSS3 a ES5. Zkráceně se ale tato trojce označuje jako technologický balíček HTML5.

Kompletní dokumentace o těchto technologiích je k dispozici na stránkách konsorcia World Wide Web Consortium W3C

Odkazy na základní značky jsou dostupné na mnoha místech a webových stránkách, odkud je čerpán i následující obsah textu[30].

[30] Další informace naleznete především na stránkách:
http://w3c.github.io/html-reference/spec.html#hgroup
dále v českém jazyce: http://www.html5.cz/, www.jakpsatweb.cz, http://programujte.com/clanek/2010082200-html5-nove-vlastnosti/, https://www.zdrojak.cz/serialy/webdesigneruv-pruvodce-po-html5/, http://www.itnetwork.cz/html-css/html-manual, http://www.fce.vutbr.cz/studium/materialy/0u2/reference_html.pdf
v anglickém jazyce: http://html5doctor.com/the-hgroup-element/, http://www.w3schools.com/

Základní značky HTML a jejich atributy

Struktura dokumentu

```
<HTML>
<HEAD> <TITLE>...</TITLE>...</HEAD>
<BODY> ... </BODY>
</HTML>
```

Záhlaví dokumentu

`<TITLE>...</TITLE>`	titulek (název) dokumentu
`<LINK>`	definování vazeb na další dokumenty
`<STYLE>...</STYLE>`	definice stylů
`<SCRIPT>...</SCRIPT>`	vložení skriptů (může být i kdekoliv v těle)

Atributy elementu <BODY>

`bgcolor`	barva pozadí ("#rrggbb" nebo název barvy)
`text`	barvy textu ("#rrggbb" nebo název barvy)
`background`	URL obrázku, který se použije na pozadí

Seznamy

`<UL>...</UL>`	seznam s odrážkami
`type`	tvar odrážek (`disc, square, circle`)
`<OL>...</OL>`	číslovaný seznam
`type`	způsob číslování (`1` – arabské číslice, `a` – malá písmena, `A` – velká písmena, `i` – malé římské číslice, `I` – velké římské číslice)
`start`	počáteční hodnota odrážky
`<LI>`	položka seznamu
`type`	stejné jako u <UL> nebo <OL>
`value`	hodnota číslice v odrážce pro položku číslovaného seznamu
`<DL>...</DL>`	seznam definic
`<DT>`	definovaný pojem
`<DD>`	text definice
`compact`	zhuštěný tvar seznamu (pro všechny druhy)

Obrázky

`<IMG src="URL">`	zařazení obrázku do textu
`align`	způsob zarovnání (`left, right, top, middle, bottom`)
`alt`	text pro znakové prohlížeče
`width`	šířka
`height`	výška
`border`	šířka rámečku
`hspace`	velikost horizontálního odsazení od textu
`vspace`	velikost vertikálního odsazení od textu

Tabulky

`<TABLE>...</TABLE>`	vymezení tabulky
`align`	způsob obtékání tabulky textem (`left, right`) nebo umístění tabulky doprostřed (`center`)
`cellspacing`	mezera mezi buňkami
`cellpadding`	odsazení obsahu buňky od rámečku
`width`	šířka tabulky v pixelech nebo procentech
`border`	šířka rámečku
`rules`	zobrazení vnitřních čar (`all, cols, rows, none`)
`frame`	způsob zobrazení vnějších obrysů tabulky (`void, above, below, hsides, vsides, lhs, rhs, border`)
`<TR>...</TR>`	vymezení řádku tabulky
`<TH>...</TH>`	buňka záhlaví tabulky
`<TD>...</TD>`	buňka tabulky
`<COLGROUP> </COLGROUP>`	vymezení skupiny sloupců
`<COL>`	specifikace atributů pro sloupce
`bgcolor`	barva pozadí buňky
`align`	vodorovné zarovnání (`left, center, right, justify`)
`valign`	svislé zarovnání (`top, middle, bottom`)
`width`	šířka buňky v pixelech nebo v procentech z celkové šířky tabulky
`nowrap`	text v buňce se nebude zalamovat
`rowspan`	počet sloučených řádků (pro TD, TH)
`colspan`	počet sloučených sloupců (pro TD, TH)
`span`	počet sloupců se společným nastavením (pro COL resp. COLGROUP)

Vybrané entity

`<`	<	`&ndash;`	– (krátká pomlčka)
`>`	>	`&mdash;`	— (dlouhá pomlčka)
`&`	&	`&ldquo;`	“
`&quot;`	"	`&bdquo;`	„
` `	nezlomitelná mezera	`&times;`	×
`&deg;`	°	`&#189;`	½

Odkazy

`<A href="URL">...</A>`	odkaz na určité URL
`title`	název odkazu
`<A name="kotva">...</A>`	definice kotvy

Logické styly písma

`<EM>...</EM>`	zvýraznění
`<STRONG>...</STRONG>`	silné zvýraznění
`<BIG>...</BIG>`	velké písmo
`<SMALL>...</SMALL>`	malé písmo
`<SUB>...</SUB>`	dolní index
`<SUP>...</SUP>`	horní index

Fyzické styly písma

`<TT>...</TT>`	neproporcionální písmo (psací stroj)
`<I>...</I>`	kurzíva
`<B>...</B>`	tučné
`<U>...</U>`	podtržené
`<FONT>...</FONT>`	definice velikosti, barvy, fontu písma
`color`	barva
`size`	velikost 1–7

Formátování textu a strukturování dokumentu

`<P>...</P>`	vymezení odstavce
`align`	způsob zarovnání (`left,right,center`)
` `	nový řádek
`clear`	vynechání místa až skončí obrázky (`left, right,all,none`)
`<HR>`	horizontální čára
`align`	způsob zarovnání (`left,right,center`)
`size`	výška v pixelech
`width`	délka v pixelech nebo v procentech
`<Hn>...</Hn>`	nadpis úrovně *n* (1–6)
`align`	způsob zarovnání (`left,right,center`)
`<PRE>...</PRE>`	předformátovaný text
`<SPAN>...</SPAN>`	vymezení části textu se společným formátováním písma
`<DIV>...</DIV>`	Označení související části dokumentu
`align`	způsob zarovnání (`left,right,center`)

Formuláře

`<FORM>...</FORM>`	vymezení formuláře
`name`	jméno formuláře (v rámci dokumentu jedinečné)
`<INPUT>`	definice vstupního resp. výstupního prvku
`type=`	druh vstupního pole
`text`	jednořádkové textové pole
`checkbox`	zaškrtávací políčko
`radio`	přepínač
`submit`	tlačítko k odeslání dat na server
`reset`	tlačítko pro obnovení hodnot ve formuláři
`button`	obecné tlačítko (pro navázání skriptu)
možné vlastnosti INPUT prvků (smysluplné podle typu):	
`name`	jméno (potřebné pro zpracování skripty)
`value`	hodnota
`checked`	nastavuje implicitní zaškrtnutí
`size`	rozměr pole ve znacích
`maxlength`	maximální počet znaků
`<TEXTAREA>...</TEXTAREA>`	víceřádkové vstupní pole
`name`	jméno (potřebné pro zpracování skripty)
`rows`	počet řádků
`cols`	počet sloupců
`<SELECT>...</SELECT>`	výběrové pole (rozbalovací nabídka)
`size`	počet viditelných voleb
`multiple`	možnost výběru více položek současně
`<OPTION>...</OPTION>`	identifikace volby ve výběrovém poli
`value`	odesílaná hodnota

Komentáře (poznámky)

`<!-- Tohle je komentář ... -->`

Barvy

"#rrggbb" v barevném modelu RGB

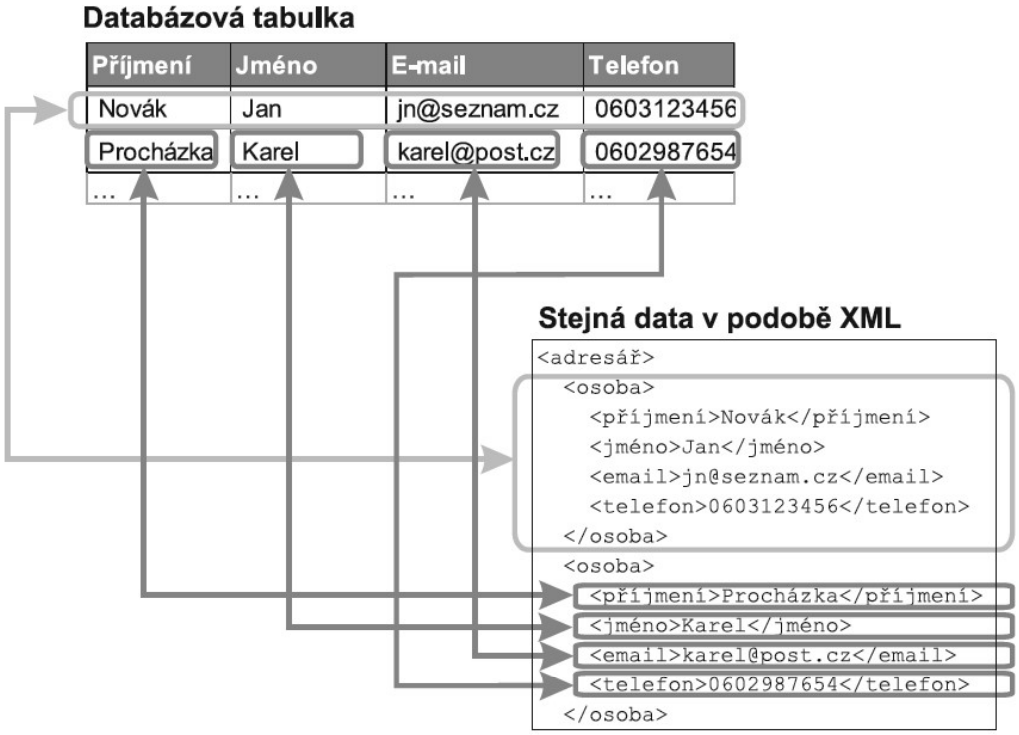
`black`	černá	`gray`	šedá
`purple`	fialová	`fuchsia`	světle fialová
`navy`	modrá	`blue`	jasně modrá
`olive`	hnědá	`yellow`	žlutá
`green`	zelená	`lime`	světle zelená
`teal`	tyrkysová	`aqua`	světle tyrkysová
`maroon`	červená	`red`	světle červená
`silver`	stříbrná (bílá)	`white`	jasně bílá

6.4.3 XML

XML (eXtensible Markup Language) je jazyk, který umožňuje označit význam jednotlivých částí textu, a ne jejich vzhled. Největší přínos XML spočívá v tom, že v dokumentech můžeme používat vlastní značky (tagy). Jedná se opět o podmnožinu SGML, oproti HTML si zachovává možnost definování vlastních DTD, a tedy i vlastních značek, pro jednotlivé skupiny dokumentů. Narozdíl od SGML je mnoho parametrů předem určeno a nelze je měnit, je dána např. maximální délka názvů značek, použité oddělovače a speciální znaky atd. XML už počítá s podporou různých jazyků, takže není tak úzce svázáno s angličtinou jako většina předchozích počítačových technologií.

Syntaxe zápisu dokumentů v XML je oproti SGML poměrně přísná, což umožňuje mnohem snazší a levnější vývoj aplikací, které umožňují s tímto jazykem pracovat.

Základní funkcí značek v XML je přidělení obsahu. Lze pomocí značky určit, že se jedná o výrobek, barvu, u dokumentu o kapitolu, odstavec, v matematice o proměnnou, vzorec apod. Poměrně obsáhle popsal práci s XML Kosek (2000).



Obrázek 62: přepis dat pomocí XML

Zdroj: XML pro každého, Kosek (2000)

6.4.4 FORMÁTOVÁNÍ V HTML, KASKÁDOVÉ STYLY

Formátováním HTML dokumentů se zabývá nepřeberné množství publikací. V prvopočátcích bylo formátování omezené na tagy HTML, později došlo k rozšíření možností pomocí samostatného jazyka CSS.

Protože se jazyk HTML vyvíjel, vznikaly časem různé způsoby, jak formátovat text. Proto dnes existují dva odlišné způsoby, jak v HTML třeba obarvit písmo nebo ztučnit text.

- Starší způsob používá přímo HTML tagy. (Například kurzíva se dělá pomocí tagů a : kurzíva). Pomocí tagů se některé věci nedají udělat.
- Novější způsob prostřednictvím tzv. CSS stylů, používá se tag <style> a obecný atribut "style", kterému se přiřazuje nějaká definice.

Důvod implementace CSS do procesu tvorby www stránek vyplynul zejména z malých možností HTML. Formátování HTML[31] bylo potřebné přizpůsobit faktu, že každý text má obsah a formu. Když mluvíme o formátu (formě) webových stránek, myslíme tím třeba barvu a velikost písma, pozadí, zarovnání atd., tj. vše, co nepatří do obsahu. Hlavním smyslem CSS je umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu. Původně to měl umožnit už jazyk HTML, ale v důsledku nedostatečných standardů a konkurenčního boje výrobců prohlížečů se vyvinul jinak. Starší verze HTML obsahují celou řadu elementů, které nepopisují obsah a strukturu dokumentu, ale i způsob jeho zobrazení. Z hlediska zpracování dokumentů a vyhledávání informací není takový vývoj žádoucí.

Obecně řečeno, CSS je nějaký zápis, který určuje vzhled (barvy, dekorační obrázky, rozmístění prvků) HTML dokumentu. V praxi se pak častěji používají tyto názvy: Kaskádové styly, styly, CSS soubory. Soubor s kaskádovými styly má příponu css. Sám o sobě nemá smysl. Jeho funkčnost se projeví až po propojení s HTML souborem.

CSS je kolekce metod pro grafickou úpravu webových stránek. Ta zkratka znamená Cascading Style Sheets, česky "kaskádové styly". Kaskádové, protože se na sebe mohou vrstvit definice stylu, ale platí jenom ta poslední.

Jazyk byl navržen standardizační organizací W3C, autorem prvotního návrhu byl Håkon Wium Lie. Byly vydány CSS1, CSS2 a CSS3.

V současnosti se dá říct, že už se celý web formátuje pomocí CSS. Ze starého HTML formátování zůstalo minimum. Pomocí CSS lze např.:

- Nastavit libovolnou a přesnou velikost písma, prokládání, kapitálky.
- Udělat odsazení prvního řádku odstavce, zvětšit řádkování.
- Zrušit nebo zvětšit prázdný prostor po odstavci.
- Automaticky formátovat nadpisy (například je všechny udělat zelené).

[31] převzato s úpravami z www.jakpsatweb.cz

- Zvýrazňovat odkazy po přejetí myší.
- Udělat automaticky grafické odrážky.
- Určité části textu zneviditelnit, zprůhlednit nebo nezobrazit.
- Předefinovat grafický význam běžných tagů (například všechno, co je kurzívou, udělat i tučně).
- Nastavit pozadí čehokoliv, stránky, tabulky ale třeba i odstavce; pozadí se nemusí opakovat a může mít přesnou pozici.
- Umístit nějaký objekt (třeba kus textu) kamkoliv do stránky, může se to i překrývat.
- Přidat k čemukoli rolovací lišty, oříznout to, orámovat, nastavit okraje.
- V kombinaci se skripty je dnes CSS nejmocnější zbraň pro "rozhýbání" stránek.
- Hlavní význam CSS spočívá v tom, že fungují hodně automaticky, přičemž se vzhled celého webu deklaruje jedním souborem.

Styl se může deklarovat třemi způsoby:

4. Přímo v textu zdroje u formátovaného elementu pomocí atributu style="...".
5. Pomocí "stylopisu" (angl. "stylesheet") v hlavičce stránky. Stylopis je jakýsi seznam stylů. Je v něm obecně napsáno, co má být jak zformátováno, například že nadpisy mají být zelené. Do stránky se stylopis píše mezi tagy <style> a </style>.
6. Použitím externího stylopisu, tj. souboru *.css, na který se stránka odkazuje tagem <link>. V souboru je umístěný stylopis. Hlavní výhoda je v tom, že na jeden takový soubor se dá nalinkovat mnoho stránek, takže pak všechny vypadají podobně.

Kaskádové styly mají jinou syntaxi než HTML. První na řadě je selektor[32] (ten určuje, co má být formátováno). V složených závorkách ({}) se nachází vlastnosti a hodnoty (jinak také blok deklarací). Zapíše se vlastnost, dvojtečka, mezera, hodnota a výčet hodnot se ukončí středníkem (;). Mezery je možné vynechávat nebo jich napsat více za sebou, funkčnost to neovlivní. Pro přehlednost je lepší mezery psát. Selektor udává, jaký element bude blok deklarací formátovat. Selektorem může být třída (class), identifikátor (id), tag, nebo různé kombinace. Identifikátoru se v CSS předřazuje křížek (#, Ctrl+Alt+X), třídám tečka a tagu se nepředřazuje nic, napíše se tak jak je (bez špičatých závorek). Je možné formátovat více elementů najednou, selektory oddělíme čárkami. Další možností je formátování vnořených elementů, pro takové formátování se selektory oddělují mezerami. Jako selektory by se neměla používat slova s diakritikou.

[32] Blíže např. na http://www.klikzone.cz/CSS-navod/pravidla-CSS.php, http://dum.hajduch.net/VY_32_INOVACE_1ICT9roc_09_B ap.

6.4.5 JAVASCRIPTY

JavaScript je multiplatformní, objektově orientovaný skriptovací jazyk, jehož autorem je Brendan Eich z tehdejší společnosti Netscape. Nyní se zpravidla používá jako interpretovaný programovací jazyk pro WWW stránky, často vkládaný přímo do HTML kódu stránky. Jsou jím obvykle ovládány různé interaktivní prvky GUI (tlačítka, textová políčka) nebo tvořeny animace a efekty obrázků.

Script v JavaScriptu

Script v JavaScriptu se obvykle spouští až po stažení WWW stránky z Internetu (tzv. na straně klienta), na rozdíl od ostatních jiných interpretovaných programovacích jazyků (např. PHP a ASP), které se spouštějí na straně serveru ještě před stažením z Internetu. Z toho plynou jistá bezpečností omezení, JavaScript např. nemůže pracovat se soubory, aby tím neohrozil soukromí uživatele. Důležité je si uvědomit, že JavaScript běží na straně klienta, všechny ty aplikace jsou tedy spouštěny v prohlížeči u uživatele. To je obrovský rozdíl oproti serverovým jazykům, jako je např. PHP. Pomocí JavaScriptu tedy můžeme měnit obsah webové stránky u uživatele, což nabízí např. tvorbu dynamických menu, různých roletek a dalších kontejnerů, které umožňují ušetřit místo na stránce když jsou zavřené a po najetí myší se otevřou. To je jistě šikovné a přehledné. JavaScript je skvělý k formátování textu, pomocí něj si můžete do napsané zprávy vkládat smajlíky nebo dokonce formátovat text jako ve Wordu. Když je na webové stránce nějaký editor, je to na 90% JavaScript. Další využití nalezneme u ukazatelů času a data a dalších efektů na webových stránkách (např. padající sníh na vánoce).

Protože odeslání stránky na server a čekání na následnou odpověď serveru je pomalé, používá se JavaScript také na validaci webových formulářů. Když například napíšete špatně email, webová stránka vás na to ještě před odesláním upozorní a není třeba stránku znovu načítat. Musíme si však uvědomit, že protože JavaScript běží na klientovi, může si ho uživatel vypnout nebo přepsat, proto nesmíme na takovouto validaci spoléhat a email musíme podruhé zkontrolovat i na serveru. Výhodou JavaScriptu je však pohodlnost a efektivnost bez zbytečných načítání stránky a prodlev. To samé platí u dříve zmíněných JavaScriptových menu, které by se i s vypnutým JavaScriptem měly zobrazit. Je jedno, že už to nebude tak hezké, protože JavaScript má 99% lidí zapnutý a některé prohlížeče ho dokonce ani neumožňují vypnout, problém je hlavně například u indexovacích robotů Googlu, který je potom nevidí, protože JavaScript nepoužívá. To by mohlo pak znamenat, že se přes menu nedostane na zbytek webu. JavaScriptové frameworky (například nejznámější JQuery, kterou používá i gigant Google, nebo Mootools), obsahují tzv. widgety, předpřipravené miniaplikace, například menu, kde si jen nastavíte, co v nich má být z zbytek za vás řeší framework a to i v případě, kdyby někdo JavaScript vypnul[33].

[33] Převzato s úpravami z http://www.itnetwork.cz/javascript/zaklady/javascript-tutorial-uvod-do-javascriptu-nepochopeny-jazyk

Syntaxe jazyka JavaScript je volně založena na syntaxi jazyka Java. Java je objektově orientovaný programovací jazyk a JavaScript může být považován jako velmi podobný jazyk.

JavaScript a Java jsou dva ovšem odlišné programovací jazyky.

Z pohledu programovacího jazyka i u JavaScriptu existují proměnné, funkce, podmínky, operátory atd. Jak plyne z názvu, programový kód je zapsán v tzv. skriptu. Celý skript je ohraničen párovým tagem script. To znamená, že tagy <script> a </script> udávají prohlížeči informaci o tom, že má mezi těmito tagy předpokládat skript. Pro vypsání textu pomocí JavaScriptu se používá funkce document.write('text').

Skripty můžete umístit:

- Do hlavičky HTML dokumentu (tj. mezi tagy <head> a </head>).
- Do těla HTML dokumentu (tj. mezi tagy <body> a </body>).
- Do externího souboru (V HTML dokumentu bude odkazováno na tento soubor).

Jestliže umístíme tento kód do HTML dokumentu, spustí se pokaždé, jakmile je dokument načten. To znamená, že se po každé aktualizaci stránky spustí znovu.

Další možností je spouštění scriptu při událostech, jako je například kliknutí na vybraný prvek. Události (Events) umožňují vložit JavaScript do HTML prvku. Jedná se tedy o speciální atributy, které se přidávají přímo do HTML prvku. Například tag h1 naformátuje vybraný text jako nadpis, vypíše daný text, atribut onClick je událostí JavaScriptu, která při kliknutí na text vyvolá příkaz „alert“ (hláška).

```
<h1 onClick="alert('Toto text hlášky');">Klikni na tento text!!!</h1>
```

Základy syntaxe JavaScriptu

Pro psaní scriptů platí několik zásad.

- Příkazy se oddělují středníkem nebo koncem řádku. V praxi se doporučuje středníky používat, protože přidávají přehlednost.
- Oproti HTML tagů záleží na velikosti písmen, ať už v názvech proměnných či v názvech objektů.
- Řetězce se uzavírají do uvozovek. Přípustné jsou i apostrofy (na české klávesnici Alt + 39). Je to jedno, zda se použije apostrofy nebo uvozovky. Někdy jsou apostrofy nutné, když potřebuji zanořovat různé druhy uvozovek.

- Pokud potřebujeme napsat nějaký speciální znak, který JavaScript interpretuje (například zmíněné uvozovky) do stránky, musí se využít tzv. escape sekvence. Před tento znak se napíše zpětné lomítko.

JavaScript zná speciální hodnotu pro pravdu: true a pro nepravdu: false. Objekty a jejich metody a vlastnosti se oddělují tečkami: objekt.podobjekt.vlastnost. Programové sekvence se uzavírají do složených závorek {} (zejména při větvení a deklaraci funkcí).

6.5 HTML

Jak bylo zmíněno dříve, v současnosti je pro tvorbu www stránek nejpoužívanější jazyk HTML5. Podle Root.cz[34] lze shrnout nové vlastnosti a možnosti následovně:

FILE API – umožňuje nahrávat neomezené množství souborů najednou, segmentaci velkého souboru na straně klienta a postupný upload těchto částí na server, kde se serverový script postará opět o jeho složení. Tím je možné naprosto obejít omezení maximálního uploadu.

FULLSCREEN API – Toto rozhraní odstraňuje nutnost používat Flash při nutnosti přepnout dokument přes celou obrazovku. To se využívá například při přehrávání videí nebo při prohlížení fotografií na sociálních sítích.

GEOLOCATION API – Pomocí tohoto rozhraní lze bez nutnosti jakýchkoliv doplňků identifikovat globální pozici. Ta je vypočítávána ze všech možných známých parametrů, na nichž závisí přesnost. Čím více údajů, tím je API přesnější. Prohlížeč se vždy nejprve zeptá, jestli chcete dané stránce sdělit své údaje o poloze.

CSS3 – je některými zdroji označován jako součást HTML5, slouží ke stylování dokumentů. Prakticky se jedná o samostatný jazyk, sloužící obecně pro aplikaci stylů ve značkovacích jazycích, tedy i v XML apod.

@FONT-FACE – Pomocí tohoto nástroje lze načíst ze serveru libovolný font a použít ho na webu. Font se načítá při načítání stránky a funguje jako standardní webové fonty Arial, Times new Roman, Verdana a další.

MEDIA API – Umožňuje přehrávat video a hudbu přímo v prohlížeči. API umožňuje naprogramovat celý přehrávač opět bez použití plug-inu Flash, bez kterého bylo přehrávání na internetu před vydáním tohoto API nemožné. Zahrnuje metody jako play(), pause(), load() a canPlayType(). Rozhraní umožňuje vytvořit vlastní prostředí přehrávače a v kombinaci s Fullscreen API i nahradit doposud využívaný Flash.

[34] HTML5: co přináší a proč se o něj zajímat, dostupné na https://www.root.cz/clanky/html5-co-prinasi-a-proc-se-o-nej-zajimat/, online 1.9.2016

TEXT TRACK API – Toto API umožňuje připojovat k přehrávaným multimédiím titulky, popisky nebo metadata.

SVG – HTML5 podporuje využití vektorové grafiky v podobě souboru .svg. Tento formát zápisu lze vytvořit ve většině vektorových editorů a lze jej snadno implementovat do HTML kódu nebo jako externí soubor svg.

CANVAS – Asi největší podporu ze strany prohlížečů má nový atribut Canvas, umožňující vykreslovat grafy pomocí svého javascriptového rozhraní. Jedná se o prvek pro generování jakékoliv grafiky pomocí JS. Jde o tzv. plátno umožňující dynamické vykreslování bitmap a jednoduché grafiky jako je tomu například v jazyce C++.

GAMEPAD API – Toto rozhraní je schopno propojit gamepad s internetovou aplikací a ovládat ji. Jde o využití především při tvorbě internetových her.

OFFLINE WEB APPLICATIONS – Do nástupu HTML5 bylo pro funkčnost internetové aplikace zapotřebí neustálé připojení k internetu. Součástí nové specifikace balíku Offline web applications je rozhraní umožňující omezenou funkčnost webové aplikace i při tzv. offline režimu, tedy v situaci, kdy nejsme připojeni k internetu. Pomocí tohoto rozhraní lze ukládat data nejprve k uživateli a až následně na server. Při přechodu do offline režimu (výpadek připojení), bychom o tato data nepřišli, na server by se odeslala až po obnově spojení.

SÉMANTIKA – HTML5 zavedlo nové tagy, které slouží pro zpřehlednění a optimalizaci celého projektu. Místo strukturování částí stránky do tzv. divů umožňuje HTML používat tagy přesně podle toho, jaká část stránky je jimi strukturována. Například máme samostatně stojící nezávislý prvek stránky, kterým může být článek, položka v e-shopu atd. Tag lze nahradit tagem <article></article>, obdobně obsah hlavičky tagem <header></header> nebo obsah patičky realizovat tagem <footer></footer>.

Přehled tagů a atributů je souhrnně definován na stránkách konsorcia W3C. Uvádíme převzatý seznam značek. Český rejstřík HTML 5 specifikace obsahuje popis všech HTML 5 tagů včetně ukázek jejich použití a vysvětlení atributů. Rejstřík, včetně odkazů, byl převzat z e-www stránek „Abecední rejstřík tagů - Český HTML 5 manuál“[35].

6.5.1 HTML STRUKTURA

base	Tag base v HTML 5 umožňuje nastavit kořenovou složku pro relativní odkazy v dokumentu.
body	Tag body v HTML 5 označuje tělo celého HTML 5 dokumentu, ve kterém nalezneme obsah celých webových stránek.

[35] Dostupné na http://www.itnetwork.cz/html-css/html-manual/html-5-abecedni-rejstrik-tagu

DOCTYPE	Tag DOCTYPE v HTML 5 sděluje prohlížeči, že kód zobrazuje HTML 5 dokument.
head	Tag head v HTML 5 označuje hlavičku dokumentu, kde můžeme specifikovat např. kódování dokumentu.
Hlavička HTML dokumentu	Popis hlavičky head HTML dokumentu a elementů do ní patřících, tedy title, style, link, base, meta, script a noscript.
html	Tag html v HTML 5 obaluje celý HTML dokument, který obsahuje hlavičku a tělo.
link	Tag link v HTML 5 se používá k provázání dokumentu s externím souborem, nejčastěji k CSS stylům.
meta	Tag meta v HTML 5 poskytuje tzv. metadata. Jedná se o informace vložené do HTML dokumentu.
Struktura HTML dokumentu	Popis struktury HTML dokumentu, tagů doctype, html, head a body.
style	Tag style v HTML 5 slouží k vložení stylování přímo do HTML dokumentu.
title	Tag title v HTML 5 obsahuje titulek stránky. Každá HTML stránka ho musí v hlavičce obsahovat.

6.5.2 TEXTOVÉ TAGY

abbr	Tag abbr v HTML 5 označuje zkratku v textu. Může se jednat o zkratku i o zkratkové slovo.
b	Tag b v HTML 5 označuje text, který se stylisticky odlišuje od ostatního textu, ale zároveň není důležitý.
cite	Tag cite se v HTML 5 používá k citování názvu díla nebo práce.
code	Tag code v HTML 5 označuje úryvek zdrojového kódu. Ve výchozím nastavení je vykreslen neproporcionálním písmem.
del	Tag del v HTML 5 slouží k označení odstraněného textu po editaci dokumentu. Element obsahuje atributy cite a datetime .
dfn	Tag dfn v HTML 5 označuje části textu, kde definujeme nějaký pojem.
em	Tag em v HTML 5 označuje část textu, která má větší význam než okolní text. Text je vykreslen kurzívou.
Frázové tagy	Mezi frázové tagy v HTML (pro označení textu) patří elementy abbr, em, strong, dfn, code, samp, kbd, var, b, i, u, s a mark.
h1	Tag h1 v HTML 5 označuje nadpis nejvyšší úrovně, který obvykle obsahuje název webových stránek.
h2	Tag h2 v HTML 5 označuje nadpis článku či podstránky. Leží vždy pod nadpisem h1.
h3	Tag h3 v HTML 5 označuje nějakou menší sekci než nadpis úrovně druhé.

h4	Tag h4 v HTML 5 označuje nějakou menší sekci než nadpis úrovně třetí.
h5	Tag h5 v HTML 5 označuje nějakou menší sekci než nadpis úrovně čtvrté. Jeho používání je však řídké.
h6	Tag h6 v HTML 5 označuje tu nejmenší možnou sekci. Téměř se nepoužívá.
i	Tag i v HTML 5 zvýrazňuje text, je vykreslován kurzívou.
ins	Tag ins v HTML 5 slouží k označení nově vloženého textu po editaci dokumentu. Obsahuje atributy cite a datetime.
kbd	Tag kdb v HTML 5 označuje text, který má být uživatelem vložen z klávesnice. Patří mezi frázové tagy.
mark	Tag mark v HTML 5 se používá při zvýraznění části citace, která je klíčová.
Nadpisy	Nadpisy slouží v HTML pro lepší orientaci a jsou klíčové pro SEO. Například Google extrahuje informace právě z nadpisů.
p	Tag p se v HTML 5 používá k rozdělování textu do odstavců. Dokument je poté přehlednější a lépe se styluje.
q	Tag q označuje krátkou řádkovou (inline) citaci. Obsahuje atribut cite.
rp	Tag rp v HTML 5 obsahuje většinou závorky, které se zobrazí v případě, že prohlížeč ruby anotace nepodporuje.
rt	Tag rt v HTML 5 označuje danou informaci o výslovnosti v tzv. ruby anotaci.
ruby	Tag ruby v HTML 5 definuje tzv. ruby anotace, které se používají v asijské typografii.
ruby, rt a rp	Tagy ruby, rt a rp se v HTML 5 používají k vložení tzv. ruby anotací do asijské typografie.
s	Tag s v HTML 5 vykresluje obsah přeškrtnutě (zpravidla označuje text, který již není aktuální nebo korektní).
samp	Tag samp v HTML 5 slouží k označení textu.
strong	Tag strong v HTML 5 označuje silnější zdůraznění než tag em. Text je vykreslen tučně.
sub	Tag sub se v HTML 5 používá k označení dolního indexu. Využijeme ho např. při psaní vzorců nebo indexaci proměnných.
sup	Tag sup se v HTML 5 používá k označení horního indexu. Využijeme ho při psaní mocnin nebo poznámek pod čarou.
u	Tag u v HTML 5 označuje podtržený text.
var	Tag var v HTML 5 slouží k označení proměnné v textu a patří mezi frázové tagy. Vykresluje se kurzívou.
wbr	Tag wbr v HTML 5 označuje místo v dlouhém slově, kde je vhodné ho zalomit.

6.5.3 LAYOUT

article	Tag article v HTML 5 označuje samotný článek. Ten může mít svojí hlavičku a i patičku.
aside	Tag aside v HTML 5 označuje postranní panel, který obvykle obsahuje doplňky k článku.
details	Tag details v HTML 5 označuje rozbalovací sekci s detaily. Pomocí tagu summary ji přiřadíme titulek.
figcaption	Tag figcaption v HTML 5 popisuje obsah v jeho rodičovském tagu figure.
figure	Tag figure v HTML 5 označuje samostatnou ilustraci, týkající se článku. Její nadpis vložíme pomocí tagu figcaption.
footer	Tag footer v HTML 5 označuje patičku stránky. Obvykle obsahuje copyright a informace o autorovi.
header	Tag header v HTML 5 označuje hlavičku stránky. Ta zpravidla obsahuje nadpis h1.
Layout (rozložení stránky)	Pro rozložení stránky do sekci (layoutu). Se používají především tagy header, hgroup, nav, footer, section, article a aside.
nav	Tag nav v HTML 5 obsahuje navigační prvky. Může být součástí header nebo stát samostatně pod ním.
section	Tag section v HTML 5 se používá zejména k označení "těla" dokumentu mezi hlavičkou a patičkou.
summary	Tag summary v HTML 5 označuje titulek sekce s detaily, kdy po kliknutí na něj se ukážou detaily.

6.5.4 TABULKY

caption	Tag caption v HTML 5 označuje nadpis tabulky, který je ve výchozím nastavení vycentrovaný.
col	Tag col v HTML 5 definuje styl sloupců ve skupině. Nachází se v rodičovském tagu colgroup.
colgroup	Tag colgroup v HTML 5 umožňuje seskupovat sloupce do skupin a těm poté nastavovat různé CSS styly.
Jednoduché tabulky	Ke tvorbě jednoduchých HTML tabulek se využívají tagy table, tr, td, th, atributy border, colspan, rowspan, headers a scope.
Pokročilé tabulky	Popis tagů k tvorbě pokročilých HTML tabulek. Tagy thead, tfoot, tbody, caption, colgroup, col a atribut span.
table	Tag table v HTML 5 označuje tabulku a obsahuje jednotlivé řádky s buňkami.
tbody	Tag tbody v HTML 5 označuje tělo tabulky, které obsahuje její data.
td	Tag td v HTML 5 označuje buňku tabulky, která může obsahovat text, obrázky a další libovolné elementy.

tfoot	Tag tfoot v HTML 5 označuje patičku tabulky, ve které je obvykle její celé shrnutí.
th	Tag th v HTML 5 označuje hlavičkovou buňku tabulky. Ve výchozím nastavení je obsah vykreslen uprostřed a je tučný.
thead	Tag thead v HTML 5 označuje hlavičku tabulky, kde je obvykle popis jednotlivých sloupců.
tr	Tag tr v HTML 5 označuje řádek tabulky a obsahuje jednotlivé buňky.

6.5.5 SEZNAMY

datalist	Tag autocomplete v HTML 5 označuje seznam možností, které jsou následně nabízeny v inputu pomocí našeptávače.
dd	Tag dd v HTML 5 označuje popis vysvětlovaného pojmu v slovníčku pojmů.
dl	Tag dl v HTML 5 označuje slovníček pojmů. Také má své vlastní tagy pro položky.
dt	Tag dt v HTML 5 označuje ve slovníčku pojmů vysvětlovaný pojem.
li	Tag li v HTML 5 označuje jednu položku seznamu a nejčastěji obaluje text.
ol	Tag ol v HTML 5 označuje uspořádaný seznam, který je řazen dle nějakého klíče.
Seznamy a slovníček pojmů	Popis tagů k tvorbě HTML seznamů a slovníku pojmů - ul, ol, li, dl, dt, dd, atributy reversed, start, type a value.
ul	Tag ul v HTML 5 označuje neuspořádaný seznam. Standardně mají jeho položky odrážky.

6.5.6 MULTIMÉDIA

area	Tag area v HTML 5 označuje jednu oblast na mapě (obrázku). Typicky se jedná o různé tvary.
audio	HTML 5 tag audio se používá pro vložení zvuků nebo hudby do HTML stránky.
canvas	Tag canvas v HTML 5 označuje plátno, na které je možné vykreslovat např. JavaScriptem. Používá se pro animace a hry.
embed	Tag embed se používá k vložení externí aplikace nebo plug-inu. Často se používá k vložení Flash aplikace.
img	Tag img v HTML 5 označuje obrázek, který má v sobě několik atributů. Je nepárový.
map	Tag map v HTML 5 umožňuje definovat mapu na obrázku přímo na straně klienta.
object	Tag object se v HTML 5 používá k vložení multimediální aplikace do HTML dokumentu. Nejčastěji se jedná o Flash.

Obrázky a obrázková mapa	Tagu img pro vložení obrázku do stránky používá atributy src, alt, width, height. Umožňuje vytvářet obrázkové klikací mapy v kombinaci s elementy map a area.
source	HTML 5 tag source se používá pro vložení zdroje videa nebo hudby pro tagy audio a video.
track	HTML 5 tag track se používá pro vložení textové stopy pro tagy audio a video.
video	HTML 5 tag video se používá pro vložení videa do HTML stránky.

6.5.7 FORMULÁŘE

button	Tag button v HTML 5 označuje tlačítko, které nemusí být bezprostředně součástí formuláře a může obsahovat další tagy.
fieldset	Tag fieldset se v HTML 5 používá s tagem legend k seskupení formulářových polí a může být označen i jako GroupBox.
Formuláře	Tag form se používá ke tvorbě webových formulářů v HTML a může obsahovat atributy method, action, name, autocomplete, accept-charset, taget a enctype.
input	Tag input se používá ve formulářích pro vložení různých vstupních polí.
input typu file	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ file slouží k nahrání souborů na web.
input typu hidden	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ hidden slouží k vložení skrytého pole.
input typu checkbox	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ checkbox slouží k vložení zaškrtávacího pole.
input typu password	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ password slouží k zadání hesla.
input typu radio	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ radio slouží k vložení přepínacího tlačítka.
input typu reset	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ reset obnoví výchozí hodnoty ve formuláři.
input typu submit	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ submit slouží k odeslání formuláře.
input typu text	Tag input se používá v HTML formulářích pro vložení různých vstupních polí. Typ text slouží k zadání krátkého textu.
keygen	Tag keygen se v HTML 5 používá k bezpečnému odeslání formuláře pomocí certifikátu.
Label	Popis tagu label, který slouží jako popisek polí v HTML formulářích. Obsahuje atributy for a form.
output	Tag output v HTML 5 označuje výsledek nějaké početní operace. Může se jednat např. o výstup JavaScriptu.
Select	Popis tagu select, který umožňuje do HTML formuláře vložit vysouvací nabídku (combo box). Souvisí s tagy option a optgroup.

textarea	Slouží ke vložení textového pole do HTML formuláře. Obsahuje atributy autofocus, cols, disabled, form, maxlength, name, placeholder, readonly.

6.5.8 OSTATNÍ TAGY

a	Tag a v HTML 5 označuje hypertextový odkaz. Pomocí něj se pohybujeme v rámci našeho webu nebo odkazujeme na web cizí.
address	Tag address v HTML 5 poskytuje kontaktní informace k dokumentu nebo jeho části. Bývá špatně chápán, neoznačuje adresu.
bdi	Tag bdi v HTML 5 označuje text, který může být psán jiným směrem textu, než text okolní.
bdo	Tag bdo v HTML 5 označuje přepsání aktuálního směru textu, např. na směr zprava doleva.
blockquote	Tag blockquote označuje blokovou citaci.
br	Tag br se používá k zalomení řádku. Můžeme tak např. zalamovat text v odstavci.
command	Tag command se v HTML 5 používá k vložení příkazu do kontextového menu nebo k zachycení klávesové zkratky ve stránce.
div	Tag div se v HTML 5 používá k seskupování logicky souvisejících blokových elementů a k jejich stylování.
hr	Tag hr v HTML 5 označuje oddělovač, který je vykreslen jako horizontální čára. Používá se při změně tématu.
iframe	Tag iframe v HTML 5 označuje inline rámec, umožňuje tedy vložení další HTML stránky do HTML dokumentu.
komentáře	Komentáře slouží v HTML pro autora stránky jako poznámky k lepší orientaci zejména v komplexním kódu.
menu	Tag menu umožňuje v HTML 5 vložit toolbar s menu, podobně jako ho známe např. z Windows aplikací.
meter	Tag meter v HTML 5 označuje měřič, podobný progressbaru, který graficky zobrazuje určitou část z celku.
progress	Tag progress v HTML 5 označuje ukazatel pokroku v nějaké činnosti, známý také jako progressbar.
script	Tagy script a noscript slouží ke vložení klientský skriptů (JavaScriptu) do HTML dokumentu. Obsahují atributy async, defer, type, charset a src.
span	Nemá žádný sémantický význam. Používá se k seskupování logicky souvisejících řádkových (inline) elementů a funguje tedy podobně, jako element .
time	HTML 5 tag time se používá pro označení data nebo času v HTML stránce.

6.5.9 ZASTARALÉ TAGY (PODPOROVANÉ Z DŮVODŮ KOMPATIBILITY)

big	Tag big v HTML dříve označoval větší text, než byl ten normální. V současnosti se větší text realizuje pomocí CSS.
center	Tag center v HTML vycentroval obsah. Nově se centruje pomocí CSS vlastností.
font	Tag font v HTML umožňoval nastavit typ, velikost a barvu písma. Nyní písmo nastavujeme v CSS dokumentu.
frame	Tag frame v HTML označoval rámec, ve kterém se zobrazovala samostatná HTML stránka.
frameset	Tag frameset v HTML označoval soubor rámců. Z HTML byly však rámce již odebrány.
noframes	Tag noframes v HTML obsahoval text, který se zobrazil, když prohlížeč nepodporoval rámce.
small	Tag small v HTML označoval a označuje menší text než ten okolní. Má stejnou velikost jako nadpis 5. úrovně.
strike	Tag strike v HTML označoval přeškrtnutý text. Nově se to dělá přes CSS vlastnost.
Stylování	Pro snazší hromadnou modifikovatelnost do HTML stránky vkládáme pouze obsah. Úprava vzhledu elementů (stylování) se vyčlenila do samostatného souboru CSS.

Jazyku HTML5 je věnován poměrně široký prostor na www stránkách, doporučuji dále studium např. https://www.zdrojak.cz/serialy/webdesigneruv-pruvodce-po-html5/, www.jakpsatweb.cz nebo https://www.zdrojak.cz/serialy/webdesigneruv-pruvodce-po-html5/

6.6 Kaskádové styly

6.6.1 CHARAKTERISTIKA CSS3

CSS3[36] – Jsou velkým přínosem pro internetové aplikace a jejich vzhled, jsou chápány jako součást HTML5 označovaná jako CSS3, která slouží pro stylování internetových dokumentů. Verze CSS3 je v podstatě průlom grafického zpracování webů. Umožňuje totiž vytvořit téměř každý grafický prvek stránky pouze pomocí tohoto stylovacího jazyka a tím velmi urychlit načítání stránek. Dalo by se zjednodušeně říct, že je tím grafika definována jako vektorová, a tedy je možné ji zobrazovat v jakýchkoliv zařízeních bez změny kvality, což je u použití obrázkové grafiky nemožné. Je tedy velmi snadné dokument přizpůsobit

[36] Převzato z https://www.root.cz/clanky/html5-co-prinasi-a-proc-se-o-nej-zajimat/

velikosti uživatelovy obrazovky. Možnosti lze shrnout pro aplikaci v následujících oblastech::

- Animace.
- Možnosti background-image (obrázkové pozadí).
- Border image (obrázkový rámeček).
- Border radius (zaoblené rohy).
- Box shadows (stíny).
- Rozměry.
- Position: Fixed (fixní pozice).
- Font settings (možnosti nastavení písma).
- Přechody.
- Hyphenation (dělení slov).
- Media queries.
- Multiple-column layout.
- Multiple backgrounds.
- Opacity (neprůhlednost).
- Pointer events.
- Text overflow.
- Text shadow (stín textu).
- Transforms 2D (2d transformace).
- Transform 3D (3d transformace).
- Transitions.

Opět, obdobně jako u specifikace HTML5 uvádím abecední rejstřík tak, jak je specifikován na www v příspěvku „Abecední rejstřík - Český CSS 3 manuál“[37].

6.6.2 PŘÍKAZY A KLÍČOVÉ VÝTAZY

Text a písmo

@font-face	Pomocí CSS 3 pravidla @font-face vytváříme vlastní font s jeho vlastním názvem.
color	Pomocí CSS 3 vlastnosti color nastavujeme barvu písma/textu v HTML elementu.
content	Pomocí CSS 3 vlastnosti content nastavujeme, co bude před a za elementem.
counter-increment	Pomocí CSS 3 vlastnosti counter-increment můžeme přičítat k číslu vlastní hodnotu a tak tvořit seznamy.

[37] Převzato z http://www.itnetwork.cz/html-css/css-manual/cesky-css-3-manual-rejstrik

counter-reset	Pomocí CSS 3 vlastnosti counter-reset definujeme ID počítadla a zároveň ho resetujeme.
direction	Pomocí CSS 3 vlastnosti direction nastavujeme směr textu v HTML elementu, tedy zleva doprava nebo zprava doleva.
font	Pomocí CSS 3 vlastnosti font nastavíme písmo v HTML elementu, styl, variantu, tloušťku, velikost, řádkováná a rodinu.
font-family	Pomocí CSS 3 vlastnosti font-family nastavujeme rodinu písma v HTML elementu. Rodinou písma se myslí font (např. Arial).
font-size	Pomocí CSS 3 vlastnosti font-size nastavujeme velikost písma/textu v HTML elementu.
font-size-adjust	Pomocí CSS 3 vlastnosti font-size-adjust nastavujeme velikost písma fontu. Podporuje pouze Mozilla Firefox.
font-stretch	Pomocí CSS 3 vlastnosti font-stretch nastavujeme, zda bude text užší nebo širší. Nepodporuje žádný webový prohlížeč.
font-style	Pomocí CSS 3 vlastnosti font-style nastavujeme styl písma v HTML elementu. Vlastnost se používá pro kurzívu.
font-variant	Pomocí CSS 3 vlastnosti font-variant nastavujeme variantu písma v HTML elementu. Vlastnost se používá pro kapitálky.
font-weight	Pomocí CSS 3 vlastnosti font-weight nastavujeme tučnost písma v HTML elementu. Vlastnost se používá pro tučné písmo.
Fonts (písma)	Praktická ukázka vytvoření a použití vlastního písma pomocí CSS3. Již žádná závislost na web-safe fonts.
hanging-punctuation	Pomocí CSS 3 vlastnosti hanging-punctuation nastavujeme, jak bude umístěné interpunkční znaménko.
letter-spacing	Pomocí CSS 3 vlastnosti letter-spacing zvyšujeme rozestoupení znaků v textu, tedy mezery mezi jednotlivými písmeny.
line-height	Pomocí CSS 3 vlastnosti line-height nastavujeme výšku řádku textu, tedy řádkování.
Multiple Columns (sloupce)	Členění textu do sloupců jako například v novinách za použití CSS3.
quotes	Pomocí CSS 3 vlastnosti quotes nastavujeme vlastní uvozovky pro citace.
tab-size	Pomocí CSS 3 vlastnosti tab-size nastavujeme velikost znaku tabulátoru. Funguje pouze pro elementy textarea a pre.
Text Effects (efekty textu)	CSS3 manuál. Ukázka práce s novými možnostmi textu a efekty, které nám CSS3 přináší. Stínování, zalamování, tvorba efektů.
text-align	Pomocí CSS 3 vlastnosti text-align stylujeme zarovnání textu v HTML elementu, např. vlevo, vpravo, na střed a do bloku.
text-align-last	Pomocí CSS 3 vlastnosti text-align-last nastavujeme zarovnání posledního řádku.
text-decoration	Pomocí CSS 3 vlastnosti text-decoration nastavujeme dekoraci textu v HTML elementu, tou se myslí např. podtržení.
text-decoration-color	Pomocí CSS 3 vlastnosti text-decoration-color nastavujeme barvu podtrhnutí.

text-decoration-line	Pomocí CSS 3 vlastnosti text-decoration-line nastavujeme textu čáru (např. podtržení).
text-decoration-style	Pomocí CSS 3 vlastnosti text-decoration-style nastavujeme styl čáry u textu.
text-indent	Pomocí CSS 3 vlastnosti text-indent nastavujeme odsazení prvního řádku odstavce, jako je to u českých odstavců.
text-justify	Pomocí CSS 3 vlastnosti text-justify nastavujeme mezery k hodnotě justify (vlastnost text-align).
text-overflow	Pomocí CSS 3 vlastnosti text-overflow nastavujeme chování vytékání textu z HTML elementu a jeho případné zkrácení.
text-shadow	Pomocí CSS 3 vlastnosti text-shadow přidáme textu stín. Docílíme tak hezkého efektu např. v nadpisech.
text-transform	Pomocí CSS 3 vlastnosti text-transform ostylujeme text tak, aby se zobrazoval pouze velkými nebo malými písmeny.
unicode-bidi	Pomocí CSS 3 vlastnosti unicode-bidi můžeme text vykreslit zrcadlově převrácený.
white-space	Pomocí CSS 3 vlastnosti white-space nastavujeme chování mezer v textu. Můžeme ponechat několik mezer vedle sebe.
word-break	Pomocí CSS 3 vlastnosti word-break nastavujeme rozdělování slov v CJK textu.
word-spacing	Pomocí CSS 3 vlastnosti *word-spacing* nastavujeme rozestupy (mezery) mezi jednotlivými slovy v textu.
word-wrap	Pomocí CSS 3 vlastnosti word-wrap nastavujeme zalamování (rozdělování) slov v textu.

Pozadí

background	CSS 3 vlastnost background slouží k ostylování pozadí HTML elementu. Umožňuje přidat vícenásobné pozadí.
background-attachment	CSS 3 vlastnost background-attachment slouží k přichycení pozadí HTML elementu jako fixní při rolování stránkou.
background-clip	CSS 3 vlastnost background-clip slouží k oříznutí pozadí HTML elementu.
background-color	CSS 3 vlastnost background-color slouží k nastavení barvy pozadí HTML elementu.
background-image	CSS 3 vlastnost background-image slouží k nastavení obrázku pozadí HTML elementu. Umožňuje přidat vícenásobné pozadí.
background-origin	CSS 3 vlastnost background-origin nastavuje, k čemu se má vztahovat pozice obrázku na pozadí HTML elementu.
background-position	CSS 3 vlastnost background-position slouží k nastavení pozice pozadí HTML elementu.
background-repeat	CSS 3 vlastnost background-repeat slouží k nastavení opakování pozadí HTML elementu.

background-size	CSS 3 vlastnost background-size slouží ke změně velikosti pozadí HTML elementu.
Backgrounds (pozadí)	CSS3 manuál. Vlastnost background. Ukázka práce s novými možnostmi, které nám CSS3 přináší. Více obrázků na pozadí, velikost, poloha zobrazení.

Pozicování a velikost

bottom	Pomocí CSS 3 vlastnosti bottom nastavujeme dolní pozici HTML elementu.
box-sizing	Pomocí CSS 3 vlastnosti box-sizing nastavujeme, co bude šířka a výška elementu zahrnovat.
clear	Pomocí CSS 3 vlastnosti clear nastavujeme, v které části elementu není povolen plovoucí obsah.
clip	Pomocí CSS 3 vlastnosti clip nastavujeme viditelnou část absolutně pozicovaných elementů.
display	Pomocí CSS 3 vlastnosti display nastavujeme jakým způsobem má být HTML element vykreslován.
float	Pomocí CSS 3 vlastnosti float nastavujeme, zda má být element plovoucí.
height	Pomocí CSS 3 vlastnosti height nastavujeme výšku HTML elementu, můžeme tak měnit jeho velikost/rozměry.
left	Pomocí CSS 3 vlastnosti left nastavujeme levou pozici HTML elementu.
max-height	Pomocí CSS 3 vlastnosti max-height nastavujeme maximální výšku HTML elementu. To je výhodné zejména u obrázků.
max-width	Pomocí CSS 3 vlastnosti max-width nastavujeme maximální šířku HTML elementu. To je výhodné zejména u obrázků.
min-height	Pomocí CSS 3 vlastnosti min-height nastavujeme minimální výšku HTML elementu.
min-width	Pomocí CSS 3 vlastnosti min-width nastavujeme minimální šířku HTML elementu.
overflow	Pomocí CSS 3 vlastnosti overflow nastavujeme chování HTML elementu v případě, že jeho obsah přeteče rozměry elementu.
overflow-x	Pomocí CSS 3 vlastnosti overflow-x nastavujeme chování elementu v případě, že jeho obsah přeteče šířku elementu.
overflow-y	Pomocí CSS 3 vlastnosti overflow-y nastavujeme chování elementu v případě, že jeho obsah přeteče výšku elementu.
position	Pomocí CSS 3 vlastnosti position nastavujeme typ pozice elementu, statickou, relativní, absolutní nebo fixní.
resize	Pomocí CSS 3 vlastnosti resize můžeme uživateli dovolit roztahovat element jako například u textarey.
right	Pomocí CSS 3 vlastnosti right nastavujeme pravou pozici HTML elementu.

top	Pomocí CSS 3 vlastnosti top nastavujeme horní pozici HTML elementu.
vertical-align	Pomocí CSS 3 vlastnosti vertical-align nastavujeme svislé zarovnání HTML elementu.
visibility	Pomocí CSS 3 vlastnosti visibility můžeme skrýt HTML element.
width	Pomocí CSS 3 vlastnosti width nastavujeme šířku HTML elementu, můžeme tak měnit jeho velikost/rozměry.
z-index	Pomocí CSS 3 vlastnosti z-index nastavujeme hloubku (z-souřadnici) HTML elementu na stránce.

Rámeček

border	Pomocí CSS 3 vlastnosti border nastavujeme rámeček okolo HTML elementu.
border-bottom	Pomocí CSS 3 vlastnosti border-bottom nastavujeme spodní hranu rámečku okolo HTML elementu.
border-bottom-color	Pomocí CSS 3 vlastnosti border-bottom-color nastavujeme barvu dolní hrany rámečku okolo HTML elementu.
border-bottom-left-radius	Pomocí CSS 3 vlastnosti border-bottom-left-radius nastavujeme poloměr zakulacení levého dolního rohu rámečku.
border-bottom-right-radius	Pomocí CSS 3 vlastnosti border-bottom-right-radius nastavujeme poloměr zakulacení pravého dolního rohu rámečku.
border-bottom-style	Pomocí CSS 3 vlastnosti border-bottom-style nastavujeme styl čáry dolní hrany rámečku okolo HTML elementu.
border-bottom-width	Pomocí CSS 3 vlastnosti border-bottom-width nastavujeme šířku (tloušťku) spodní hrany rámečku okolo HTML elementu.
border-color	Pomocí CSS 3 vlastnosti border-color nastavujeme barvu rámečku okolo HTML elementu.
border-image	Pomocí CSS 3 vlastnosti border-image nastavíme obrázek jako rámeček okolo HTML elementu.
border-image-outset	Pomocí CSS 3 vlastnosti border-image-outset nastavujeme, jak daleko od obsahu divu má být obrázkový rámeček.
border-image-repeat	Pomocí CSS 3 vlastnosti border-image-repeat nastavujeme, jestli se bude obrázkový rámeček opakovat.
border-image-slice	Pomocí CSS 3 vlastnosti border-image-slice nastavujeme, jak se obraz rámečku uřízne.
border-image-source	Pomocí CSS 3 vlastnosti border-image-source nastavujeme adresu k obrázku rámečku.
border-image-width	Pomocí CSS 3 vlastnosti border-image-width nastavujeme šířku rámečku s obrázkem.
border-left	Pomocí CSS 3 vlastnosti border-left nastavujeme levou hranu rámečku okolo HTML elementu.
border-left-color	Pomocí CSS 3 vlastnosti border-left-color nastavujeme barvu levé hrany rámečku okolo HTML elementu.

border-left-style	Pomocí CSS 3 vlastnosti border-left-style nastavujeme styl čáry levé hrany rámečku okolo HTML elementu.
border-left-width	Pomocí CSS 3 vlastnosti border-left-width nastavujeme šířku (tloušťku) levé hrany rámečku okolo HTML elementu.
border-radius	Pomocí CSS 3 vlastnosti border-radius nastavujeme poloměr zakulacení rohů rámečku okolo HTML elementu.
border-right	Pomocí CSS 3 vlastnosti border-right nastavujeme pravou hranu rámečku okolo HTML elementu.
border-right-color	Pomocí CSS 3 vlastnosti border-right-color nastavujeme barvu pravé hrany rámečku okolo HTML elementu.
border-right-style	Pomocí CSS 3 vlastnosti border-right-style nastavujeme styl čáry pravé hrany rámečku okolo HTML elementu.
border-right-width	Pomocí CSS 3 vlastnosti border-right-width nastavujeme šířku (tloušťku) pravé hrany rámečku okolo HTML elementu.
border-style	Pomocí CSS 3 vlastnosti border-style nastavujeme styl čáry rámečku okolo HTML elementu.
border-top	Pomocí CSS 3 vlastnosti border-top nastavujeme horní hranu rámečku okolo HTML elementu.
border-top-color	Pomocí CSS 3 vlastnosti border-top-color nastavujeme barvu horní hrany rámečku okolo HTML elementu.
border-top-left-radius	Pomocí CSS 3 vlastnosti border-top-left-radius nastavujeme poloměr zakulacení levého horního rohu rámečku.
border-top-right-radius	Pomocí CSS 3 vlastnosti border-top-right-radius nastavujeme poloměr zakulacení pravého horního rohu rámečku.
border-top-style	Pomocí CSS 3 vlastnosti border-top-style nastavujeme styl čáry horní hrany rámečku okolo HTML elementu.
border-top-width	Pomocí CSS 3 vlastnosti border-top-width nastavujeme šířku (tloušťku) horní hrany rámečku okolo HTML elementu.
border-width	Pomocí CSS 3 vlastnosti border-width nastavujeme šířku rámečku okolo HTML elementu.
Borders (rámečky)	CSS3 manuál. Vlastnost border. Použití zakulacených rámečků, přidání vlastního stínu, použití obrázku jako rámečku.
outline	Pomocí CSS 3 vlastnosti outline nastavujeme obrys okolo rámečku HTML elementu.
outline-color	Pomocí CSS 3 vlastnosti outline-color nastavujeme barvu obrysu okolo HTML elementu.
outline-style	Pomocí CSS 3 vlastnosti outline-style nastavujeme styl čáry obrysu okolo HTML elementu.
outline-width	Pomocí CSS 3 vlastnosti outline-width nastavujeme šířku obrysu okolo HTML elementu.

Okraje (margin a padding)

margin	Pomocí CSS 3 vlastnosti margin nastavujeme okraj elementu, vzdálenost mezi rámečkem a okolím elementu.

margin-bottom	Pomocí CSS 3 vlastnosti margin-bottom nastavujeme dolní okraj elementu, vzdálenost mezi dolní hranou rámečku a okolím.
margin-left	Pomocí CSS 3 vlastnosti margin-left nastavujeme levý okraj elementu, vzdálenost mezi levou hranou rámečku a okolím.
margin-right	Pomocí CSS 3 vlastnosti margin-right nastavujeme pravý okraj elementu, vzdálenost mezi pravou hranou rámečku a okolím.
margin-top	Pomocí CSS 3 vlastnosti margin-top nastavujeme horní okraj elementu, vzdálenost mezi horní hranou rámečku a okolím.
padding	Pomocí CSS 3 vlastnosti padding nastavujeme vzdálenost mezi rámečkem a obsahem elementu.
padding-bottom	Pomocí CSS 3 vlastnosti padding-bottom nastavujeme vzdálenost mezi dolní hranou rámečku a obsahem elementu.
padding-left	Pomocí CSS 3 vlastnosti padding-left nastavujeme vzdálenost mezi levou hranou rámečku a obsahem elementu.
padding-right	Pomocí CSS 3 vlastnosti padding-right nastavujeme vzdálenost mezi pravou hranou rámečku a obsahem elementu.
padding-top	Pomocí CSS 3 vlastnosti padding-top nastavujeme vzdálenost mezi horní hranou rámečku a obsahem elementu.

Seznamy a tabulky

border-collapse	Pomocí CSS 3 vlastnosti border-collapse stylujeme rámeček HTML tabulky tak, aby se vykresloval jako jednoduchá čára.
border-spacing	Pomocí CSS 3 vlastnosti border-spacing nastavujeme rozestupy mezi rámečky okolo buněk HTML tabulky.
caption-side	Pomocí CSS 3 vlastnosti caption-side nastavujeme umístění nadpisu HTML tabulky.
empty-cells	Pomocí CSS 3 vlastnosti empty-cells nastavujeme, zda se má skrýt rámeček a pozadí prázdných buněk HTML tabulky.
list-style	Pomocí CSS 3 vlastnosti list-style nastavujeme styl odrážek seznamů.
list-style-image	Pomocí CSS 3 vlastnosti list-style-image nastavíme obrázek odrážkám seznamu.
list-style-position	Pomocí CSS 3 vlastnosti list-style-position nastavíme pozici odrážek seznamu.
list-style-type	Pomocí CSS 3 vlastnosti list-style-type nastavíme typ odrážek seznamu. Máme na výběr různé tvary, číslované i abecední.
table-layout	Pomocí CSS 3 vlastnosti table-layout nastavujeme algoritmus výpočtu šířky sloupců HTML tabulky.

Ostatní vlastnosti

@keyframes	Pomocí CSS 3 pravidla @keyframes nastavujeme jméno animace a v jeho obsahu chování animace.

@media	Pomocí CSS 3 pravidla @media nastavujeme rozdílné styly pro různá zařízení či různé média typy.
2D Transforms (transformace)	Libovolné natáčení, škálování, posun či natahování libovolného HTML elementu pomocí CSS3.
3D Transforms (transformace)	Libovolné natáčení, škálování, posun či natahování libovolného HTML elementu ve třetím rozměru pomocí CSS3.
align-content	Pomocí CSS 3 vlastnosti align-content nastavujeme vertikální pozicování ohebných elementů.
align-items	Pomocí CSS 3 vlastnosti align-items nastavujeme pozici ohebných sloupců.
align-self	Pomocí CSS 3 vlastnosti align-self nastavujeme pozici pouze daného ohebného sloupce.
animation	Pomocí CSS 3 vlastnosti animation si můžeme vytvářet vlastní animace pomocí pouhého CSS.
animation-delay	Pomocí CSS 3 vlastnosti animation-delay nastavujeme dobu před začátkem animace.
animation-direction	Pomocí CSS 3 vlastnosti animation-direction nastavujeme jakým směrem (a rychlostí) se animace přehraje.
animation-duration	Pomocí CSS 3 vlastnosti animation-duration nastavujeme dobu trvání animace.
animation-fill-mode	Pomocí CSS 3 vlastnosti animation-fill-mode nastavujeme, co se děje s elementem, když je animace pozastavená.
animation-iteration-count	Pomocí CSS 3 vlastnosti animation-iteration-count nastavujeme, kolikrát se bude animace opakovat.
animation-name	Pomocí CSS 3 vlastnosti animation-name nastavujeme jméno animace.
animation-play-state	Pomocí CSS 3 vlastnosti animation-play-state nastavujeme, zda je animace zastavená nebo má běžet.
animation-timing-function	Pomocí CSS 3 vlastnosti animation-timing-function nastavujeme rychlostní křivku animace.
Animations (animace)	Tvorba animací pomocí CSS3.
backface-visibility	Pomocí CSS 3 vlastnosti backface-visibility nastavujeme viditelnost zadní strany elementu.
box-shadow	Pomocí CSS 3 vlastnosti box-shadow nastavujeme stín pod HTML elementem.
column-count	Pomocí CSS 3 vlastnosti column-count nastavujeme počet sloupců, do kterých se text rozdělí.
column-fill	Pomocí CSS 3 vlastnosti column-fill nastavujeme, jak se sloupce budou naplňovat.
column-gap	Pomocí CSS 3 vlastnosti column-gap nastavujeme velikost mezery mezi sloupci.
column-rule	Pomocí CSS 3 vlastnosti column-rule nastavujeme pravidla stylů mezi sloupci.

column-rule-color	Pomocí CSS 3 vlastnosti column-rule-color nastavujeme barvu stylu mezi sloupci.
column-rule-style	Pomocí CSS 3 vlastnosti column-rule-style nastavujeme styl čar mezi sloupci.
column-rule-width	Pomocí CSS 3 vlastnosti column-rule-width nastavujeme šířku čar mezi sloupci.
column-span	Pomocí CSS 3 vlastnosti column-span nastavujeme, jak bude daný element roztažený napříč všemi sloupečky.
column-width	Pomocí CSS 3 vlastnosti column-width si můžeme rozdělit text do několika sloupců podle zadané šířky.
columns	Pomocí CSS 3 vlastnosti columns si můžeme rozdělit text do několika sloupců.
cursor	Pomocí CSS 3 vlastnosti cursor nastavujeme, jaký kurzor myši se má zobrazit nad HTML elementem.
flex	Pomocí CSS 3 vlastnosti flex nastavujeme velikost pružného elementu od ostatních.
flex-basis	Pomocí CSS 3 vlastnosti flex-basis nastavujeme základní velikost pružného elementu.
flex-direction	Pomocí CSS 3 vlastnosti flex-direction nastavujeme směr ohebných elementů.
flex-flow	Pomocí CSS 3 vlastnosti flex-flow nastavujeme směr a chování ohebných elementů při zaplnění toho rodičovského.
flex-grow	Pomocí CSS 3 vlastnosti flex-grow nastavujeme část velikosti rodičovského elementu pružnému elementu.
flex-shrink	Pomocí CSS 3 vlastnosti flex-shrink zmenšujeme pružný element, když všechny elementy jsou větší než ten rodičovský.
flex-wrap	Pomocí CSS 3 vlastnosti flex-wrap nastavujeme, zda se při zaplnění velikosti rodičovského elementu jeho potomci seřadí.
justify-content	Pomocí CSS 3 vlastnosti justify-content nastavujeme horizontální pozicování ohebných elementů.
nav-down	Pomocí CSS 3 vlastnosti nav-down definujeme posunutí označení šipkou dolů.
nav-index	Pomocí CSS 3 vlastnosti nav-index nastavujeme sekvenční navigační pořadí elementu.
nav-left	Pomocí CSS 3 vlastnosti nav-left definujeme posunutí označení šipkou doleva.
nav-right	Pomocí CSS 3 vlastnosti nav-right definujeme posunutí označení šipkou doprava.
nav-up	Pomocí CSS 3 vlastnosti nav-up definujeme posunutí označení šipkou nahoru.
opacity	Pomocí CSS 3 vlastnosti opacity nastavujeme průhlednost HTML elementu. Ukážeme si i jak na průhledné pozadí.
order	Pomocí CSS 3 vlastnosti order nastavujeme pořadí ohebných elementů.

page-break-after	Pomocí CSS 3 vlastnosti page-break-after nastavujeme zalomení stránky po elementu při tisku stránky.
page-break-before	Pomocí CSS 3 vlastnosti page-break-before nastavujeme zalomení stránky před elementem při tisku stránky.
page-break-inside	Pomocí CSS 3 vlastnosti page-break-inside nastavujeme zalomení stránky v elementu při tisku stránky.
perspective	Pomocí CSS 3 vlastnosti perspective nastavujeme vzdálenost mezi osou Z a uživatelem.
perspective-origin	Pomocí CSS 3 vlastnosti perspective-origin měníme 3D elementu pozici dolních bodů.
transform	Pomocí CSS 3 vlastnosti transform můžeme element různě transformovat (natáčet, zkosit, posouvat, natahovat atd.).
transform-origin	Pomocí CSS 3 vlastnosti transform-origin můžeme měnit pozici transformovaného elementu.
transform-style	Pomocí CSS 3 vlastnosti transform-style určujeme vnoření elementu v 3D prostoru.
transition	Pomocí CSS 3 vlastnosti transition můžeme vytvářet různé přechodné efekty např. pro změnu barvy elementu.
transition-delay	Pomocí CSS 3 vlastnosti transition-delay nastavujeme dobu před začátkem přechodného efektu.
transition-duration	Pomocí CSS 3 vlastnosti transition-duration nastavujeme dobu trvání přechodného efektu.
transition-property	Pomocí CSS 3 vlastnosti transition-property nastavujeme, co se bude měnit v přechodném efektu.
transition-timing-function	Pomocí CSS 3 vlastnosti transition-timing-function nastavujeme rychlostní křivku přechodného efektu.
Transitions (přechody)	Použití přechodového efektu při změně vlastnosti elementu.
User interface (rozhraní)	Nové vlastnosti pro změny velikosti elementu, resizování boxů, outline.

6.7 JavaScript

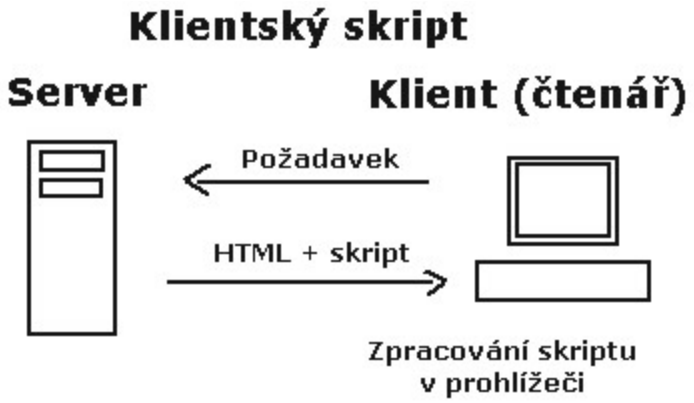
Jak bylo zmíněno, JavaScript (JS) umožňuje vložit akci na straně klienta.[38] Využití na straně klientské aplikace je podmíněno schopností HTML kódu „rozumět“ příkazům scriptů (ve smyslu přečtení příkazů) a schopností prohlížeče interpretovat příkazy JS. V praxi je prohlížečem zpracováván HTML kód do chvíle, než je nalezen příkaz pro předání řízení JavaScriptu. Následně jsou zpracovávány příkazy JS, s tím souvisí akceptace syntaxe JS (v době zpracovávání příkazů JS neplatí syntaxe HTML ale syntaxe JS). JS je zpracováván do chvíle, než je předáno opět řízení HTML.

[38] Předpokládáme standardní klientskou aplikaci, nikoli serverovou

Vzhledem k poměrně rozsáhlé problematice, vychází tato kapitola z textu publikovaném na www.jakpsatrweb.cz, text je převzatý s drobnými úpravami, doporučuji důkladné prostudování uvedených stránek, které současně obsahují množství příkladů.

6.7.1 Úvod do JavaScriptu

JavaScript je programovací jazyk, který se používá v internetových stránkách. Zapisuje se přímo do HTML kódu, což je velká výhoda, protože je to jednoduché. JavaScript je klientský skript. To znamená, že se program odesílá se stránkou na klienta (do prohlížeče) a teprve tam je vykonáván, jak je zřejmé z obrázku 42. (Protikladem klientských skriptů jsou skripty serverové, které jsou vykonávány na serveru a na klienta jdou už jen výsledky.) Existují i jiné jazyky klientských skriptů, například VBScript. JavaScript je často zaměňován s Javou. Java je samostatný programovací jazyk. Má s JavaScriptem pouze podobnou syntaxi. Pro práci s JS je doporučeno zvládnutí základů HTML a základ programování.



Obrázek 63: Zpracování JS – klientský script

Zdroj:www.jakpsatweb.cz

6.7.2 Charakteristiky jazyka

JavaScript – charakteristika jazyka

- Interpretovaný -- nemusí se kompilovat.
- Objektový -- využívá objektů prohlížeče a zabudovaných objektů.
- Závislý na prohlížeči -- funguje ve většině prohlížečů.
- Case sensitivní -- záleží na velikosti písem v zápisu.
- Syntaxí podobný jazykům C, Java a podobným.

Omezení jazyka

- JavaScript funguje pouze v prohlížeči.
- Uživatel může JavaScript zakázat.

- Existují různé odlišné verze jazyka i prohlížečů, což vede k častým chybám.
- Neumí přistupovat k souborům (kromě cookies) ani k žádným systémovým objektům.
- Neumí žádná data uložit (kromě cookies).

Po zvládnutí základů je nejlepší všímat si cizích skriptů na cizích stránkách. Většina skriptů je zapsána přímo ve zdrojovém kódu stránky, takže se dají zkopírovat.

6.7.3 OBJEKTOVÝ MODEL

JavaScript je jazyk objektový, třebaže nevyužívá všechny možnosti OOP (objektově orientovaného programování). 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. Studium JavaScriptu je vlastně studium objektového modelu prohlížeče a dokumentu; je třeba se naučit, jak se které prvky zapisují.

Zápis

K adresování objektů se používá tečková syntaxe objektů (v jiných jazycích se používají např. šipky ->). Většina objektů má podobjekty nebo vlastnosti nebo metody; syntaxe je objekt.podobjekt, objekt.vlastnost nebo objekt.metoda().

Metoda objekt.metoda() je sama o sobě příkazem, který něco dělá. Má za sebou závorky. Vlastnost objekt.vlastnost nic nedělá, ale má hodnotu. Hodnota se dá číst nebo zapisovat, některé vlastnosti jsou jen pro čtení, některé jen pro zápis. Podobjekt objekt.podobjekt může mít další metody, vlastnosti a podobjekty. (V literatuře se podobjekt v některých případech považuje také za vlastnost.)

Přístup k objektům

- K objektům okna prohlížeče (window).
- Přes okno prohlížeče k prvkům stránky (window.document), které mají úzkou vazbu na jazyk HTML.
- K zabudovaným objektům (Date, Math, string).
- K vytvořeným objektům (to nepatří do objektového modelu a použije se to zřídka).

Nekompatibilita prohlížečů

Základní úskalí práce s JavaScriptem spočívá v tom, že objektové modely jednotlivých prohlížečů se liší. Některé objekty existují jenom v některých prohlížečích. V praxi je tedy třeba různými podmínkami testovat verzi prohlížeče a na základě toho skript větvit.

Práce s objekty

Stávající přehled není úplným výčtem, autor stránek www.jakpsatweb.cz je však považuje za prioritní, proto je uvádím i ve skriptech.

- Objekt window.
- Metody objektu window.
- Objekt window.event.
- Objekt document.
- Objekt String.
- Objekt Date.
- Objekt Math.

Dále se budeme věnovat zejména objektu window, s dalšími objekty doporučuji se seznámit na uváděných www stránkách.

6.7.4 OBJEKT WINDOW

Disponuje podbjekty: location - history - navigator - screen - frames - event - document - ostatní vlastnosti. Objekt window (okno) je vrcholem hierarchie objektů, se kterými může JavaScript pracovat. Celý objektový model (kromě Date, Math, String) je vlastně seznam podobjektů objektu window.

Pro zjednodušení lze při zápisu u podobjektů (vlastností) „window“ vynechávat. Např. zápisy window.navigator a navigator jsou ekvivalentní.

6.7.5 VLASTNOSTI A METODY OBJEKTU WINDOW.

Objekt window.location

Nebo jenom location. Je to adresa načteného dokumentu.
Načtení nového dokumentu:
window.location.href = "http://www.opf.slu.cz"; načte do okna domovskou stránku OPF.
Aktualizace stránky reload():
location.reload(false); aktualizuje stránku, pokud byla změněna,
location.reload(true); natáhne stránku ze serveru, i když se nezměnila
location.replace("http://www.seznam.cz"); natáhne stránku Seznamu, ale současnou stránku nezařadí do historie.
Zjištění adresy dokumentu (například načtením do proměnné adresa)
adresa = location.href;
Zjištění fragmentu adresy, například jméno domény:
domena = location.hostname;

Pokud by adresa byla třeba http://www.seznam.cz/cokoliv/, tak by se do proměnné vložila hodnota "www.seznam.cz".

Objekt window.history

Historie prohlížení stránek. Dovoluje vracet se ke dříve zobrazeným stránkám. Obsahuje několik metod, žádné vlastnosti ke čtení. Kvůli tomu jsou možnosti tohoto objektu velmi omezené. Pro spolehlivou složitější práci s historií je nutno ukládat informace do cookies nebo zkusit vlastnost document.referrer.

Metody objektu history:

history.back(); načte do prohlížeče minulou stránku. Je to to samé, jako když uživatel klikne na tlačítko Zpět v prohlížeči.

history.back(3); o tři stránky zpět. Jako trojí kliknutí. Když se číslo neuvede (minulý příklad), je tam jako jednička.

history.forward(2); o dvě stránky vpřed. Většinou je to nepoužitelné, protože metoda forward je přístupná jedině když se předtím zacouvalo.

history.go(-1); je také návrat na minulou stránku. Číslo v argumentu může být libovolné celé číslo. Metodou go() se dají nahradit back() a forward().

OBJEKT WINDOW.NAVIGATOR

Slouží ke zjišťování informací o typu a verzi prohlížeče.
navigator.appName vrací jméno aplikace prohlížeče
navigator.appVersion vrací verzi prohlížeče
navigator.javaEnabled() vrací informace o podpoře Java appletů

Většina autorů používá window.navigator pro větvení programu: když je window.navigator Internet Explorer, provede se skript optimalizovaný pro IE, když je to Netscape, provede se skript pro Netscape. Další vlastnosti objektu navigator najdete na zmiňovaných www stránkách.

Objekt window.screen

Objekt slouží pro zjišťování vlastností obrazovky.

screen.height, screen.width – zjistí výšku a šířku pracovní plochy grafického systému počítače.

screen.availWidth, screen.availHeight - zjistí dostupnou velikost plochy. Prakticky jde o totéž jako width a height, pouze výška bývá poněkud menší, pokud je stále zobrazen hlavní panel systému (šedá lišta dole). Ve skriptech doporučuji používat raději tyto vlastnosti.

screen.colorDepth – udává barevnou hloubku, počet bitů připadajících na jednu barvu. (Systémové nastavení obrazovky.)

Objekt window.event

Pomocí tohoto objektu lze nastavovat události myši a události klávesnice

Události myši

event.button - Vrací hodnotu podle stisknutých tlačítek (1 levé, 2 pravé, 4 prostřední).
event.clientX, event.clientY - vrací polohu myši vzhledem k levému hornímu rohu okna dokumentu.
event.screenX, event.screenY - vrací polohu myši vzhledem k levému hornímu rohu obrazovky.
event.offsetX, event.offsetY - souřadnice myši vzhledem k pozicovanému objektu (vrstvě).

Události klávesnice

event.altKey, event.ctrlKey, event.shiftKey - vracejí true / false (pravda nebo nepravda) podle toho, jestli je stisknutý alt, control resp. shift.
event.keyCode - vrací ascii kód stisknuté klávesy.
event.type – je vlastnost***, v***rací typ události, která nastala.
event.returnValue – vlastnost, mění předdefinované chování události
event.srcElement - vrací prvek, na němž událost nastala.
event.fromElement, event.toElement - vracejí prvky, ze kterého a na který myš při události jela. Vztahují se na vlastnosti onMouseOver a onMouseOut.

Objekt window.document

Objekt document je nejdůležitějším objektem, ke kterému JavaScript přistupuje. Document v sobě obsahuje všechno, co je nějakým způsobem spojeno s aktuální stránkou. Přes objekt document se přistupuje k obrázkům, formulářům, odkazům, k barvám atd. K objektům dokumentu přistupujeme proto, aby se mohly měnit. Například lze zaměnit obrázek, dá se změnit hodnota v políčku formuláře, spustit hudba, dají se měnit barvy dokumentu atd. To vše v reakci na uživatelské události nebo ještě při načítání stránky.

V praxi není důležité, že to jsou všechno operace na objektu document. Projevuje se to jenom zápisem, který vždy začíná "document. ...".

K jednotlivým prvkům na stránce se dá přistupovat pomocí objektů document.images , document.forms, document.applets, document.link, document.anchors, document.all, document.frames, document.styleSheets, document.scripts, document.selection a dalších.

K jednotlivým prvkům stránky se dá přistupovat více způsoby. Z nich nejzajímavější je metoda ***document.getElementById().*** Dokáže zpřístupnit skriptu libovolný prvek na stránce, který má nastavené id="řetězec". Další podobné metody pro zpřístupnění prvků vždy začínají ***getElementBy***, případně ***getElementsBy***.

Otatní vlastnosti objektu window

window.defaultStatus - ovlivní text ve stavové řádce
window.status - je skoro totéž, ale vázáno dočasně na událost
window.closed - vrací true, pokud je okno zavřeno (nešlape v IE 3)
window.opener - vrací odkaz (ukazatel) na okno, které jej otevřelo
window.lenght - vrací počet rámů v okně nebo rámu
window.name - vrací jméno rámu nebo okna
window.parent - vrací odkaz na nadřazený rám (okno)
window.self - vrací sebe samo

window.top - vrací odkaz na hierarchicky nejvyšší rám

window.offscreenBuffering - když se nastaví true, okno se načte, přepočítá a teprve potom zobrazí.

6.7.6 METODY

Metoda znamená, že se provede nějaká akce (na rozdíl od vlastnosti)

window.open() - otevře nové okno. Nejčastěji se používá pro vyskakovací reklamy v nových okénkách, protože umožňuje vypnout zobrazení tlačítek a předepsat přesné umístění a velikost okna.

window.showModalDialog() – umožní otevírat i okna, která čekají na uzavření a teprve potom umožňují vrátit se do původního okna.

window.close() – umožní zavření okna., buďto aktuálního, nebo zavření okna, které bylo dříve otevřeno metodou ***window.open*** a jeho identifikátor uložen do proměnné.

moveBy() - posouvá oknem o souřadnice.

moveTo() - umístí levý horní roh okna na přesnou souřadnici obrazovky.

resizeBy() - zmenší nebo zvětší velikost okna.

resizeTo() - změní velikost okna na přesně zadané velikosti v pixelech.

scrollBy() - odroluje dokument podle zadaných souřadnic (horizontálně, vertikálně). Nahoru se roluje zápornou druhou hodnotou.

scrollTo() - odroluje dokument na přesnou pozici (horizontální, vertikální). Alternativou je metoda ***scroll()***.

V systému Windows je vždy jen jedno okno aktivní (má focus), ostatní jsou v pozadí.

window.blur() přenese okno do pozadí, deaktivuje okno. Aktivním se stane jiné okno.

window.focus() je opakem blur(). Přenese okno do popředí (zamodří mu proužek). Akce se musí provádět z jiného okna (většinou po příkazu window.open) nebo pomocí časování.

Pro jednoduchou formu komunikace pomocí dialogových oken lze použít metody alert(), prompt() a confirm().

window.alert("Text hlášky") - jednoduchý dialog, zobrazí text ze závorky.

proměnná = ***window.prompt("zadej hodnotu", "přednastavený text") -*** načte uživatelem zadaný text do proměnné.

confirm() zobrazí dialog s textem a s tlačítkem OK a Storno. Do proměnné se uloží true nebo false (pravda nebo nepravda) podle toho, co uživatel zmáčknul.

print() - vytiskne aktuální okno.

window.setTimeout(), window.clearTimeout(), window.setInterval(),
window.clearInterval() - používají se na časování událostí.

6.7.7 ZÁKLADNÍ DATOVÉ TYPY, PROMĚNNÉ, FUNKCE, OPERÁTORY

Jak bylo zmíněno dříve, JavaScript je plnohodnotný programovací jazyk. Jako takový používá standardní nástroje, proměnné, operátory a funkce. Proměnnou deklarujeme klíčovým slovem ***var*** následovaným výpisem použitých proměnných. JS nevyžaduje deklaraci, proměnné se inicializují při prvním přiřazení hodnoty. Není třeba deklarovat typ proměnné (číslo, text apod.), JavaScript určí typ, JS provádí automatickou konverzi při použití. Základní datové typy jsou tyto String (text), Number (číslo s/bez desetinné čárky), Boolean

(pravda/nepravda), objekt a null nebo undefined. S proměnnými lze provádět standardní výpočty. Vzhledem k tomu, že JS je jazyk case-senzitivní, je proměnná x je různá od proměnné X. Všechny proměnné typu text musejí mít hodnoty zadané v uvozovkách nebo v apostrofech. Apostrofy je nutno použít, pokud se celý skript nachází v uvozovkách (in-line zápis).

Např.: <a onmouseover="myska='prejeta' ">. Stejně jako u CSS jsou dostupné escape sekvence, to znamená, že znak, který je používaný jako klíčový (např. uvozovky, závorky apod.) lze zadávat pomocí znaku zpětné lomítko.

Např: promenna = "Obsah";

6.7.8 PROMĚNNÉ

String

String, řetězec, je řada znaků uložená v za sebou jdoucích bajtech paměti. Maximální délka není přesně specifikována, záleží na interpretu. Proměnné typu řetězec mají funkce a vlastnosti, mezi nejčastěji používané patří:

Index - za pomoci indexů lze procházet jednotlivé znaky v řetězci stejně jako v poli. Čísluje se od nuly.

Length - vlastnost vrací délku řetězce ve znacích.

Trim - odstraní specifické znaky (mezery apod.) okolo řetězce, aby nekomplikovali další zpracování.

Replace - funkce nahradí hledanou hodnotu v řetězci jinou.

toUpperCase a toLowerCase - změní všechna písmena v řetězci na velká nebo na malá:

Concat - spojuje dva a více řetězců. Tato funkce je volána automaticky když se pro spojení omylem

Substring a Substr - funkce vrací vybranou část řetězce, které se říká podřetězec. Obě funkce dělají v podstatě to samé, avšak liší se významem parametrů.

Split - rozdělí řetězec na pole řetězců pomocí určitého znaku.

IndexOf, LastIndexOf a Search - vrací pozici daného podřetězce.

Match - vrací shodu s výrazem v řetězci. Používá se hlavně pro regulární výrazy (Regex).

Číslo

Čísla lze zapsat dvěma způsoby, standardně nebo pomocí vědecké (inženýrské, exponenciální) notace, *var x = 10; var x = 10e5; // 10**5 = 100 000.*

S čísly lze provádět většinu známých základních operací jako sčítání, násobení, dělení, existují však i složitější výpočetní operace, např. zbytek po dělení (tzv. modulo). Všechna čísla jsou v JavaScriptu ukládána jako 64-bit double a jsou počítána s přesností na 15 čísel.

Obdobně, jako u textových řetězců, i u čísel existují funkce a vlastnosti čísel.

isNaN(hodnota) - zjistí, zda je objekt v parametru funkce číslo či nikoli. Vrací true nebo false (pravda/nepravda) podle toho, zda je parametrem číslo. NaN Označuje zkratku Not a Number.

toPrecision(x) - ořízne číslo na danou přesnost.

i.toString() - převede číslo na plnohodnotný řetězec znaků.

toExponential(x) - změní zápis čísla na exponenciální (vědeckou) notaci.

Number.MAX_VALUE a Number.MIN_VALUE - vrací největší / nejmenší možné číslo.

Boolean

Logický datový typ, nabývá pouze dvou hodnot 0-1 (true-false). Jeho hodnota je tedy buď pravdivá, nebo nepravdivá. Používá se např. pro vyhodnocení podmínek a cyklů. Hodnot se zapisují bez uvozovek.

Null / Undefined

Oba dva datové typy zastupují prázdnou proměnnou. NULL je nevytvořený objekt, UNDEFINED naopak objektem není.

6.7.9 FUNKCE

Funkce je v podstatě blok kódu, který jednou napíšeme a potom ho můžeme libovolně volat bez toho, abychom ho psali znovu a opakovali se. Funkci deklarujeme pomocí klíčového slova ***function*** a obsahuje blok kódu ve složených závorkách.

V případě potřeby zpracování segmentu kódu deklarovaného jako funkce, funkci tzv. zavoláme. Toto lze provést až po tom, co funkci deklarujeme, jinak by ji prohlížeč neznal.

Funkce s parametry

Funkce může mít libovolný počet vstupních parametrů, které píšeme do závorky v její definici a podle nich ovlivňujeme její chování.

Syntaxe deklarace funkce v JavaScriptu:
function jmenoFunkce(parametry) {tělo funkce};
nebo podrobněji zapsáno:
function jmenoFunkce(parametr, parametr)
{
příkaz; příkaz; return hodnota
};

Volání funkce:

jmenoFunkce(hodnota, hodnota);
Velmi často se funkce volají na základě událostí dokumentu přímo z HTML kódu, například:
Obsah

Při kliknutí na slovo "Obsah" se vyvolá funkce upozorneni() s hodnotou parametru "hlavní stránka". Předtím samozřejmě musí být funkce inicializovaná.

Pokud funkce vrací hodnotu (deklarace obsahuje return hodnota), dá se funkce volat zápisem

proměnná = jmenoFunkce(parametry);

Proměnné ve funkci

Proměnná deklarovaná ve funkci klíčovým slovem ***var*** je lokální. Lokální proměnné jsou i parametry funkce (to, co je v závorce za jménem funkce). Pokud se ve funkci použije jméno jiné nedeklarované proměnné, jde o proměnnou globální. Výhoda funkcí je tedy v přehlednosti a úspornosti (můžeme napsat nějakou věc jednou a volat ji vícekrát na různých místech skriptu). Když se rozhodneme funkci změnit, provedeme změnu jen na jednom místě a tato změna se projeví všude, což značně snižuje riziko chyb.

Uložení funkce do proměnné

JavaScript se liší od jiných jazyků tím, jak pracuje s funkcemi. Funkce zde umí ještě více, funkci můžeme totiž uložit do běžné proměnné a z této proměnné ji později volat. Všechny funkce v JavaScriptu jsou vnitřně proměnné. Funkci můžeme definovat přímo v přiřazení do proměnné, hovoříme potom o tzv. anonymní funkci

Zabudované funkce JavaScriptu

JavaScript obsahuje předdefinované funkce. Tyto funkce se dají chápat také jako metody objektu window, které se dědí na všechny rodičovské elementy.

eval - vychází slova vyhodnotit (evaluate). Funkce vezme svoje argumenty a vyhodnotí je jako by to byl kus programu. Používá se zejména pro dynamickou změnu kódu. Tuto funkci mají v oblibě všichni programátoři JavaScriptu, protože umožňuje zápis konstrukcí běžných z vyšších jazyků, které by JavaScript jinak nevzal.

escape a unescape - funkce escape() umí zakódovat řetězce pro přenos. Rozkódování se provede funkcí unescape(). Používá se zejména v souvislosti s cookies a předávání parametrů z formulářů. Jde o to, aby při přenosu nenastaly problémy např. s diakritikou a speciálními znaky.

isFinite - funkce vrátí false, pokud je číslo v argumentu nekonečné. To se může stát třeba při dělení nulou nebo při počítání logaritmu z nuly:

isNaN - vrací true (pravdu), pokud argument není číslo, false pokud je číslo.

parseFloat a parseInt - převodní a zaokrouhlovací funkce. Argumentem je řetězec, který je převeden na číslo. ***parseInt*** z něj vrátí jenom celou část, ***parseFloat*** i s desetinou částí. Funkce ***parseInt*** má parametry dva, první je co se parsuje, druhý je zaklad (2 až 36).

6.7.10 OPERÁTORY

Operátory vycházejí z jazyka Java resp. C. Kromě níže uvedených existují ještě bitové operátory, které se prakticky nepoužívají.

Operátory přiřazení

Slouží k nastavení hodnot proměnných. Standardním operátorem je =, ale používají se i jiné operátory.

= přiřazení; += přičtení, ale také připojení řetězce ; *= , -=, /= přinásobení, odečtení, přidělení; ++ přičtení 1; -- odečtení 1.

Obdobný význam mají i méně využívané operátory, <<=, >>= , >>>=, &=, ^= a |=,

Početní operátory

+ sčítání, spojování řetězců; - odčítání, unární negace; * násobení; / dělení.

Logické operátory

== rovnost (dvě rovnítka); != nerovnost; <,<=,>=,> aritmetické srovnání; && logické AND (a zároveň); || logické OR (nebo); ! logické NOT (negace); ? : podmínkový výběr (ternární operátor), logické spojení (třeba v zápisu parametrů funkcí).

Logické operátory se používají zejména při větvení programů na stanovení podmínek. Podmínky nabývají hodnot true (pravda) a false (nepravda). Je potřebné si uvědomit, že rovnost je realizována dvěma rovnítky (jedno rovnítko znamená přiřazení).

6.7.11 VĚTVENÍ

Základem každého programového kódu je využívání opakujících se sekvencí. JS, stejně jako jiné programy, umožňuje větvení programu a definování cyklů. Základní způsoby větvení a cyklů v JavaScriptu jsou příkazy ***if, while*** a ***for***.

if(podmínka) {program} - jestliže je splněna podmínka, vykonává se program
while(podmínka) {program} - program se opakuje, dokud trvá podmínka
for(iniciace; podmínka; navýšení){program} cyklus (vykonávaný většinou přesně několikrát)

Jako podmínka se musí uvést výraz, jehož logická hodnota je true nebo false (pravda nebo nepravda).

Je-li v těle podmínky If jeden příkaz, nemusí se uzavírat do složených závorek. Také lze vynechat else, není-li potřeba. Časté chyby, se kterými se u if můžeme setkat, jsou: špatný zápis podmínky (pro rovnost se musejí používat dvě rovnítka), chybějící středníky nebo složené závorky, nadbytečné středníky, mezery většinou nehrají v zápisu roli.

Rozhodovací operátor ?

Tzv. ternální operátor, umožňuje rychlejší zápis rozhodování, pokud chci pouze přiřadit hodnotu proměnné. Syntaxe příkazu je ***proměnná = podmínka ? hodnota1 : hodnota2***;

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

Přepínač

switch case – slouží pro větvení do více alternativ. Syntaxe příkazu je:

```
switch(proměnná) {
    case hodnota : příkaz; break;
    case hodnota2 : příkaz2; break;
    ...
    default : příkazx;
}
```

While – lze použít ve dvou tvarech, jako cyklus s podmínkou na začátku, nebo s podmínkou na konci**.** Cyklus s podmínkou na začátku provádí sekvenci příkazů (uzavřených ve složených závorkách, následujících za while) tak dlouho, dokud platí podmínka. Jakmile podmínka neplatí, vnitřek příkazu ***while*** se už nevykoná a program bude pokračovat pod koncem sekvence ***while***. Podmínka na konci má tvar ***do{sekvence příkazů}while (podmínka)***

V souvislosti s cykly je ještě nutné zmínit příkaz ***break,*** který předčasně ukončí cyklus ***while*** nebo ***for*** a příkaz ***continue,*** který skočí na začátek cyklu.

Uváděný přehled příkazů je redukovaná množina, sloužící k nástinu vlastností a možností JavaScriptu. Doporučuji pro pochopení odzkoušet příklady vztahující se k textu na stránkách jakpsatweb.cz. Z těchto stránek byly většinou převzaty i texty vztahující se k Css a JavaScriptům.

SHRNUTÍ KAPITOLY

V této kapitole jste se seznámili s principy hypertextu a jeho použití pro publikování WWW stránek. Dozvěděli jste se jak stránky formátovat a jak vložit dynamiku do stránek.