
# Úvod do XHTML

**Ivo Šnábl**

**Web studio Institut biostatistiky a analýz MU**

snabl@iba.muni.cz

*Operační program Vzdělávání pro konkurenceschopnost*
*Projekt Zvyšování IT gramotnosti zaměstnanců vybraných fakult MU*
*Registrační číslo: CZ.1.07/2.2.00/15.0224, Oblast podpory: 7.2.2*

# Tutoriály pro Photoshop

## *V češtině*

http://photoshop.apple-mac.cz/ - především lekce 1, 2, 5, 6

http://www.tutorials.cz/clanek1248-zaklady-v-photoshopu-cs3-prvni-spusteni-lekce-v-ovladani#clanky
podrobnější lekce:

- Úvod do Photoshopu CS3 - Seznámení s programem
- Základy práce s perem
- Základy v Photoshopu CS3 - První spuštění, lekce v ovládání
- Základy v Photoshopu CS3 - Zkoušíme tvořit, práce s vrstvami
- Základy v Photoshopu CS3 - Manipulace s pracovní plochou
- Základy v Photoshopu CS3 - Práce se selektivními nástroji všeho druhu
- Základy v Photoshopu CS3 - Manipulace s předměty, libovolná transformace a jiné
- Základy v Photoshopu CS3 - Styly vrstev
- Základy v Photoshopu CS3 - Text v Photoshopu
- Základy v Photoshopu CS3 - Používání štětců a nástroje Rozmazání
- Základy v Photoshopu CS3 - Nahrávání nových doplňků (štětce, fonty, vlastní tvary)
- Základy v Photoshopu CS3 - Zkratky
- Základy v Photoshopu CS3 - Práce s maskami
- Základy v Photoshopu CS3 - Světlost, kontrast, barevnost
- Základy v Photoshopu CS3 - Zjednodušte si tvorbu

## *Vybrané anglické tutoriály*

Zmenšení velikosti
http://www.photoshopessentials.com/essentials/resizing-vs-resampling.php

Korekce a retuše
http://www.twistedtreephoto.com/Photoshop%20tutorial/Photoshop%20tutorial1.html http://www.youtube.com/watch?v=lJsJlgztd5U

## *Offline zdroje*

- součástí krabicové verze CS4 jsou i CD/DVD s tutoriály a výukovými lekcemi, v angličtině

## *Další zdroje pro rozšířené studium Photoshopu*

http://www.grafika.cz/photoshop/

**Plugin pro tvorbu ikonek ve formátu ICO**
http://www.telegraphics.com.au/sw/

# K procvičení:

- vytvořte favicon.ico pro web (rozměry 16 x 16 px, plná nebo omezená barevná paleta – která bude lepší?)
- vytvořte průhlednou verzi favicon.ico
- všechny verze využijte u několika testovacích stránek a prohlédněte výsledek v různých prohlížečích

- jak ovlivňovat indexování vyhledávači – optimální alternativa k META ROBOTS je vytvoření a uložení malého txt souboru robots.txt, více viz. http://www.jakpsatweb.cz/robots-txt.html

- klíčová slova a META DESCRIPTION: vytvořte a sestavte pro jednu vybranou stránku webu

- sestavte XHTML kód obsahující hlavní menu webu, včetně popisků TITLE u odkazů, využijte i několik alternativních klíčových slov. Menu by mělo mít oddělovníky mezi odkazy + mělo by být součástí jednoho bloku (zkuste popř. formátovat: centrované menu, s barevným pozadím, v absolutní vrstvě pomocí CSS position). Pomocí inline tagu span naformátujte (libovolně) odlišně jednu položku z menu, tak aby znázorňovala vybranou tj. aktuální stránku.

- vytvořte stránku s jednoduchým kontaktním formulářem a zvalidujte jej.

- optimalizujte obrázek s certifikátem (viz. naposledy zaslané ukázky) uložením ve Photoshopu, a pomocí relativní adresy vložte znovu do stránky. Obrázek vybavte popisem TITLE a správným pojmenováním souboru.

- upravte 2 - 3 fotografie pro web (např. foto vašich provozoven) – úprava velikosti (jednotné rozměry !), popř. jasu a kontrastu, … a uložení jako optimalizovaný JPG. Pro každou fotografii vytvořte zmenšeninu o velikosti 150 x 100 px, vložte do stránky a vybavte odkazy na optimalizované fotografie (ty vytvořené na začátku).

Níže: posledně probíraný text + doplnění o 3 kapitoly věnované tabulkám.

## Procvičení po nastudování tabulek:

- vytvořte layout podle vlastního návrhu s pomocí tabulek.

Problematiku barvy a obrázků na pozadí je možno zatím vynechat, v tuto chvíli jde zejména o:

- použití tabulek v XHTML
- validní formátování, především ohledně rozměrů (šířka tabulky: 100%, poměrné šířky sloupců; problém s prázdnými buňkami v IE6; jak simulovat chybějící atribut height?)

# Metadata a hlavička pro XHTML dokument

Velkou výhodou (X)HTML dokumentů je, že kromě vlastního obsahu, čitelného v prohlížeči, mohou nést i řadu skrytých informací, metadat. Ve skutečnosti nám mohou metadata prozradit mnohem více než dokument sám. V poslední době však už bývají hlavičky, kam se metadata zapisují, tak objemné, že se v nich lze jen těžko orientovat a někdy svým rozsahem dokonce podstatně převyšují dokument sám. Tento článek by vám měl napomoci zorientovat se v základních možnostech využití metadat v hlavičce (X)HTML dokumentu.

Nejprve se podívejte, jak bude vypadat výsledný dokument. Jednotlivé sekce budu rozebírat a komentovat postupně, protože je jich velké množství. Každá sekce je od ostatních oddělena vnořenými komentáři, které nesou souhrn těch nejdůležitějších informací. (Komentáře jsou v angličtině, protože vznikaly v rámci volně přístupného fóra součinností několika přispěvatelů.) Věnujte pozornost především elementům "meta" a "link", které obsahují metadata, ostatní elementy jsou v ukázce zahrnuty pouze proto, aby výsledkem byl validní XHTML dokument.

```
<!-- ?xml version='1.0' encoding='windows-1250'? -->
<!DOCTYPE html
PUBLIC '-//W3C//DTD XHTML 1.0 Strict//EN' 'http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd'>
<html xmlns='http://www.w3.org/1999/xhtml' xml:lang='cs' lang='cs'>
   <head>
      <!-- ENCODING /-->
      <!-- use proper one: iso-8859-2, windows-1250, utf-8 etc. /-->
      <!-- check FIRST LINE with xml charset too!!! /-->
      <meta http-equiv='Content-Type' content='text/html; charset=windows-1250' />
      <meta http-equiv='Content-language' content='cs' />
      <!-- ENCODING end /-->

      <!-- CACHE /-->
      <!-- turning cache on/OFF depends on what you need to do /-->
      <!-- dynamic (PHP/ASP) should be OFF, static (HTM/HTML) should be ON /-->
      <!-- CACHE - MSIE /-->
      <meta http-equiv='Cache-Control' content='must-revalidate, post-check=0, pre-check=0' />
      <meta http-equiv='Pragma' content='public' />
      <!-- CACHE - MSIE end /-->
      <!-- CACHE - other browsers /-->
      <meta http-equiv='Cache-Control' content='no-cache' />
      <meta http-equiv='Pragma' content='no-cache' />
      <meta http-equiv='Expires' content='-1' />
      <!-- CACHE - other browsers end /-->
      <!-- CACHE - end /-->

      <!-- ROBOTS /-->
      <!-- select suitable behavior for robots /-->
      <meta name='robots' content='index,follow' />
      <meta name='googlebot' content='index,follow,snippet,archive' />
      <!-- meta name='robots' content='noindex,nofollow' /-->
      <!-- meta name='googlebot' content='noindex,nofollow,nosnippet,noarchive' /-->
      <!-- ROBOTS end /-->

      <!-- KEYWORDS & CATEGORIES - but who cares now :-( /-->
      <meta name='description' content='XHTML 1.0 Strict basic page fragment' />
      <meta name='keywords' content='XHTML 1.0, CSS, test' />
      <meta name='category' content='XHTML' />
      <!-- KEYWORDS & CATEGORIES - end /-->

      <!-- AUTHOR self promo - use 'crypted' e-mails defeats robotic harvesters /-->
      <!-- Translate . => &#45; @ => &#64; etc. /-->
      <meta name='author' content='All: Author Name, e-mail: author-name(@)example.net' />
      <meta name='webmaster' content='All: Webmaster Name, e-mail: webmaster-name(@)example.net' />
      <meta name='copyright' content='©2004 Owner Name, e-mail: owner-name(@)example.net' />
      <!-- AUTHOR self promo - end /-->

      <!-- PICS label - content rating & description (kids, security...) /-->
      <!-- Platform for Internet Content Selection (PICS) - http://www.w3.org/PICS/ /-->
      <!-- PICS label generator - http://www.icra.org/_en/label/extended/ /-->
      <meta http-equiv='pics-label' content='(pics-1.1 "http://www.icra.org/ratingsv02.html" comment
"ICRAonline EN v2.0" l gen true for "http://example.net" r (nz 1 vz 1 lz 1 oz 1 cz 1)
"http://www.rsac.org/ratingsv01.html" l gen true for "http://example.net" r (n 0 s 0 v 0 l 0))' />
      <!-- PICS label - end /-->

      <!-- DUBLIN CORE /-->
      <!-- DC Metadata Initiative (DCMI) - http://dublincore.org /-->
      <!-- DC template generator - http://www.lub.lu.se/cgi-bin/nmdc.pl /-->
      <meta name='DC.Title' content='Title Name' />
```

```
      <meta name='DC.Identifier' content='http://www.example.net' />
      <meta name='DC.Language' content='cs' />
      <!-- DUBLIN CORE - end /-->

      <!-- GEOURL /-->
      <!-- Geographics location (altitude, latitude) - http://geourl.org /-->
      <!-- You can obtain exact location via http://www.maporama.com/share/ /-->
      <meta name='ICBM' content='0.0000, 0.0000' />
      <!-- GEOURL - end /-->

      <!-- BROWSER SPECIFIC FEATURES = ALL OFF /-->
      <!-- MSIE - 'helpful' features /-->
      <meta http-equiv='imagetoolbar' content='no' />
      <meta http-equiv='MSThemeCompatible' content='no' />
      <meta name='MS.LOCALE' content='cs' />
      <!-- OPERA - image resizing /-->
      <meta name='autosize' content='off' />
      <!-- BROWSER SPECIFIC FEATURES = end /-->

      <!-- ICON /-->
      <!-- Use icon for this page if you have one /-->
      <link rel='shortcut icon' type='image/x-icon' href='favicon.ico' />
      <!-- ICON end /-->

      <!-- ALTERNATIVE VERSION - plain text, PDA version or RSS if you have one /-->
      <link rel='alternate' type='application/rss+xml' title='RSS Example.net'
href='http://www.example.net/rss.xml' />
      <!-- ALTERNATIVE VERSION - end /-->

      <!-- NAVIGATION - based on logical relations of documents /-->
      <!-- homepage /-->
      <link rel='home' href='http://www.example.net' />
      <!-- searchpage /-->
      <link rel='search' href='http://www.example.net/search/' />
      <!-- title page of serial /-->
      <link rel='up' href='http://www.example.net/serial/this/' />
      <!-- first part of serial /-->
      <link rel='first' href='http://www.example.net/article/first/' />
      <!-- last part of serial /-->
      <link rel='last' href='http://www.example.net/article/last/' />
      <!-- previous part of serial /-->
      <link rel='prev' href='http://www.example.net/article/previous/' />
      <!-- next part of serial /-->
      <link rel='next' href='http://www.example.net/article/next/' />
      <!-- news, RSS /-->
      <link rel='newsfeed' href='http://www.example.net/rss.xml' />
      <!-- NAVIGATION - end /-->

      <!-- CASCADING STYLE SHEETS /-->
      <!-- INPAGE STYLE - pagemargin problem solving /-->
      <style type='text/css' media='all'>
         BODY, HTML {
            border: 0px none;
            margin: 0px;
            padding: 0px;
         }
      </style>
      <!-- LINKED STYLE /-->
      <link rel='stylesheet' type='text/css' media='all' href='./css/css_all.css' />
      <link rel='stylesheet' type='text/css' media='screen' href='./css/css_screen.css' />
      <link rel='stylesheet' type='text/css' media='print' href='./css/css_print.css' />
      <!-- PREFFERED STYLE /-->
      <link rel='stylesheet' type='text/css' href='./css/css_pref.css' title='Preffered style' />
      <!-- ALTERNATE STYLE /-->
      <link rel='alternate stylesheet' type='text/css' href='css_alt.css' title='Alternate style' />
      <!-- CASCADING STYLE SHEETS - end /-->

      <title>
         XHTML 1.0 Strict page basic fragment
      </title>
   </head>

   <body>
      Put XHTML 1.0 Strict page content here...
   </body>
</html>
```

## Kódování dokumentu

Bezesporu nejrozšířenější metainformací ze všech je údaj o kódování dokumentu, respektive o použité znakové sadě, a jazyku, v němž je dokument psán. Tyto dva řádky kódu, které se současný webdesign

snaží již opustit, umožnily skutečný rozvoj WWW tím, že uživatelům internetu zpřístupnily dokumenty v jejich národním jazyku. Myslím, že jsou natolik známé, že nemá smysl je zde podrobně rozebírat. Zájemce o další informace mohu odkázat na seriál Znakové sady v praxi nebo článek XHTML - kódování dokumentu. Pouze připomenu, že určení kódování a jazyka dokumentu musí vždy předcházet jakýmkoli dalším částem dokumentu a že kódování není třeba uvádět u dokumentů užívajících UTF.

```
      <!-- ENCODING /-->
      <!-- use proper one: iso-8859-2, windows-1250, utf-8 etc. /-->
      <!-- check FIRST LINE with xml charset too!!! /-->
      <meta http-equiv='Content-Type' content='text/html; charset=windows-1250' />
      <meta http-equiv='Content-language' content='cs' />
      <!-- ENCODING end /-->
```

## Kešování dokumentu

Pravidla pro kešování (uchovávání lokálních kopií) jsou další samozřejmou součástí (X)HTML dokumentu. Nemohou v plném rozsahu nahradit hlavičky HTTP protokolu, mohou však posloužit tam, kde není k HTTP hlavičkám přístup, ať už jde o bezpečnostní servery, různé proxy a nebo prohlížeče, které si HTTP hlavičky vykládají po svém. Tyto metainformace můžeme použít tehdy, když chceme donutit prohlížeč, aby vždy poskytoval čtenáři aktuální verzi dokumentu (pokud ji například generujeme dynamicky), nebo naopak když chceme trochu ulehčit internetu tím, že proxyserverům umožníme poskytovat lokální kopie dokumentu, což snižuje požadavky na přenos dat.

```
      <!-- CACHE /-->
      <!-- turning cache on/OFF depends on what you need to do /-->
      <!-- dynamic (PHP/ASP) should be OFF, static (HTM/HTML) should be ON /-->
      <!-- CACHE - MSIE /-->
      <meta http-equiv='Cache-Control' content='must-revalidate, post-check=0, pre-
check=0' />
      <meta http-equiv='Pragma' content='public' />
      <!-- CACHE - MSIE end /-->
      <!-- CACHE - other browsers /-->
      <meta http-equiv='Cache-Control' content='no-cache' />
      <meta http-equiv='Pragma' content='no-cache' />
      <meta http-equiv='Expires' content='-1' />
      <!-- CACHE - other browsers end /-->
      <!-- CACHE - end /-->
```

Předchozí zápis striktně zakazuje vytváření lokálních kopií dokumentu a nutí všechny programy k revalidaci (znovunačtení) při každém požadavku na zobrazení dokumentu (opačného efektu bychom dosáhli vymazáním bloku nebo přepisem hodnot na jejich logický opak). Všimněte si, že zde definujeme pravidla pro kešování vlastně dvakrát. Je tomu tak proto, že prohlížeče řady Microsoft Internet Explorer ignorují normy, popisující princip a funkci cache (viz RFC 2068 a navazující).

Výbornou pomůckou pro praktickou aplikaci kešování je Caching Tutorial for Web Authors and Webmasters.

## Informace pro vyhledávače a jiné roboty

Metaelement "robots" informuje o tom, jak by si autor dokumentu přál, aby se k jeho dílu chovaly programy, specializované na shromažďování metadat nebo dokumentů jako takových. V této kategorii jsou podstatné především vyhledávače, bez kterých bychom už dnes prakticky nebyli schopni na internetu cokoli najít. Tyto metaelementy umožňují nastavovat pravidla chování pro každý dokument zvlášť, je-li to potřeba, roboty je však nemusí respektovat (což zhusta činí downloadery). Rozhodně tímto způsobem nelze chránit důvěrná data! Podrobné informace získáte nejlépe na The Web Robots Pages.

```
      <!-- ROBOTS /-->
      <!-- select suitable behavior for robots /-->
      <meta name='robots' content='index,follow' />
      <meta name='googlebot' content='index,follow,snippet,archive' />
      <!-- meta name='robots' content='noindex,nofollow' /-->
```

```
      <!-- meta name='googlebot' content='noindex,nofollow,nosnippet,noarchive' /-->
      <!-- ROBOTS end /-->
```

Kódové slovo "index" znamená, že si robot může zapamatovat URL a další potřebné informace o stránce, "follow" zase značí, že robot smí následovat hypertextové odkazy v dokumentu. Opačného chování lze dosáhnout připojením záporky "no" před kódové slovo, přičemž lze vytvářet libovolné kombinace (tedy klidně i "noindex, follow" nebo "index, nofollow").

Povšimněte si, že kromě metaelementu "robots" se v této sekci vyskytuje také metaelement "googlebot". Ten je zde speciálně kvůli vyhledávači Google (i když některé další roboty ho také respektují) a umožňuje ovlivnit jeho chování pomocí dvou rozšiřujících kódových slov. Slovo "archive" znamená, že si prohlížeč může uložit snímek stránky do své lokální paměti a později ho poskytovat k nahlédnutí (což často nevyhovuje poskytovatelům placeného obsahu, třeba novin), slovo "snippet" určuje, jak bude zobrazovat výsledek hledání, jehož součástí je indexovaný dokument - v tomto případě se zobrazí kousek textu okolo hledaného slova, v opačném případě by se zobrazil jen popisek dokumentu z metaelementu "description".

## Popis, klíčová slova a kategorie dokumentu

Metaelementy této sekce měly původně sloužit pro udržování přehledu o obsahu dokumentů, po vzoru bibliografických katalogů, takže by bylo možno nalézt relevantní dokumenty i tehdy, pokud by nemluvily přímo o předmětu zájmu čtenáře. První vyhledávače jim přikládaly velkou váhu, takže je lidé brzy začali zneužívat a vkládali do nich různé nesmysly. Některé současné vyhledávače je proto zcela ignorují a jiné je berou v úvahu jen jako podpůrnou informaci. Svůj význam si tak uchovávají především v rozlehlých intranetových sítích.

```
      <!-- KEYWORDS & CATEGORIES - but who cares now :-( /-->
      <meta name='description' content='XHTML 1.0 Strict basic page fragment' />
      <meta name='keywords' content='XHTML 1.0, CSS, test' />
      <meta name='category' content='XHTML' />
      <!-- KEYWORDS & CATEGORIES - end /-->
```

Metaelement "description" by měl obsahovat lidsky srozumitelný a výstižný popis dokumentu, nepřesahující 256 znaků. Metaelement "keywords" by měl obsahovat takzvaná klíčová slova, což jsou termíny, které mají pro daný dokument mimořádný význam. V současnosti se nedoporučuje používat více než čtyři či pět slov. Metaelement "category" klasifikuje dokument podle mezinárodního předmětového katalogu a v současnosti se takřka vůbec nepoužívá.

## Autor, webmaster a vlastník dokumentu

Tato skupina metaelementů určuje vztah dokumentu k různým fyzickým či právnickým subjektům, typicky k autorovi obsahu, jeho vlastníkovi a správci webu. Informace jsou to bezesporu zajímavé a důležité, problém je pouze v tom, že jsou internetovou veřejností hromadně ignorovány. Protože však většinou obsahují e-mailové adresy, staly se lákavou pochoutkou pro různé roboty, které je shromažďují do spamovacích databází. Pokud už je v dokumentu budete uvádět, je vhodné standardní tvar e-mailové adresy nějakým způsobem rozrušit tak, aby zůstala srozumitelná pro člověka, ale stala se nečitelnou pro robota. Problém je ovšem v tom, že spamboti jsou čím dál inteligentnější, což o lidech, bohužel, nelze říci...

```
      <!-- AUTHOR self promo - use 'crypted' e-mails defeats robotic harvesters /-->
      <!-- Translate . => &#45; @ => &#64; etc. /-->
      <meta name='author' content='All: Author Name, e-mail: author-
name(@)example.net' />
      <meta name='webmaster' content='All: Webmaster Name, e-mail: webmaster-
name(@)example.net' />
      <meta name='copyright' content='©2004 Owner Name, e-mail: owner-
name(@)example.net' />
      <!-- AUTHOR self promo - end /-->
```

## Přístupnost a obsahová vhodnost dokumentu

Platform for Internet Content Selection (PICS) vznikla jako fórum, jehož cílem původně bylo ochránit nezletilé osoby (rozuměj děti) před nevhodným obsahem, například pornografií nebo "politicky nekorektním" jazykem. PICS je založeno na spolupráci mezi poskytovatelem obsahu, který své stránky zhodnotí a vnese do nich patřičné

metainformace, výrobcem programů pro přístup k obsahu (prohlížečů), který poskytne potřebné nástroje, a odpovědnou osobou (rodičem), která povolí či zakáže přístup k určitým typům dokumentů. Z logických důvodů PICS dokonale selhal, je však vhodné ho uvádět, protože některé programy považují jeho nepřítomnost za známku nekalého obsahu.

```
      <!-- PICS label - content rating & description (kids, security...) /-->
      <!-- Platform for Internet Content Selection (PICS) - http://www.w3.org/PICS/
/-->
      <!-- PICS label generator - http://www.icra.org/_en/label/extended/ /-->
      <meta http-equiv='pics-label' content='(pics-1.1
"http://www.icra.org/ratingsv02.html" comment "ICRAonline EN v2.0" l gen true for
"http://example.net" r (nz 1 vz 1 lz 1 oz 1 cz 1)
"http://www.rsac.org/ratingsv01.html" l gen true for "http://example.net" r (n 0 s 0
v 0 l 0))' />
      <!-- PICS label - end /-->
```

Jakmile autoři PICS seznali, že tudy cesta nevede, snažili se přejít od destrukce ke konstrukci. V současnosti, jak tvrdí, není účelem PICS omezovat přístup k informacím, ale naopak informace popisovat, třídit a definovat jejich zdroje (autory a podobně). Obávám se však, že v této oblasti nemohou uspět, protože jim konkuruje řada kvalitnějších a inteligentnějších systémů bez pošramocené pověsti. Pokud chcete PICS používat, doporučuji vám on-line generátor, protože vytvořit potřebný metaelement "ručně" je prakticky nemožné.

## Dublin Core Metadata Initiative

Takzvaná Dublin Core Metadata Initiative (DCMI) je v současnosti otevřeným fórem, které se snaží sbírat, systematizovat, vytvářet a poskytovat specifická schémata (sady metaelementů), umožňující popis libovolného aspektu dokumentu. Původně tato iniciativa vznikla mezi odborníky na bibliografii, jako prostředek, který měl umožnit popis dokumentů stejným způsobem, jaký byl používán ve specializovaných knihovnických programech. Naštěstí se do věci včas vložily i jiné subjekty, takže se zaměření iniciativy podstatně rozšířilo a v současnosti představuje nejrozsáhlejší a nejpropracovanější systém, pomocí něhož je možno (X)HTML dokumenty popisovat.

```
      <!-- DUBLIN CORE /-->
      <!-- DC Metadata Initiative (DCMI) - http://dublincore.org /-->
      <!-- DC template generator - http://www.lub.lu.se/cgi-bin/nmdc.pl /-->
      <meta name='DC.Title' content='Title Name' />
      <meta name='DC.Identifier' content='http://www.example.net' />
      <meta name='DC.Language' content='cs' />
      <!-- DUBLIN CORE - end /-->
```

Metaelementy DCMI snadno poznáte podle povinné předpony "DC" a naleznete je především v odborných dokumentech na specializovaných serverech. Je jich velké množství, takže je nejlepší používat pro analýzu i tvorbu specializované nástroje, jako je například Dublin Core Metadata Template. Pokud máte zájem o další informace z této oblasti a nevládnete jazykem anglickým právě nejlépe, doporučuji vám Dublin Core Czech - vliv knihovníků je zde však mnohem patrnější než na mezinárodních stránkách, takže se nedejte zmást.

## Zeměpisná poloha zdroje (serveru)

Už v šerém dávnověku internetu vznikl příznak, který umožňoval určit přesnou geografickou polohu fyzického zdroje konkrétní informace, přesněji řečeno serveru systému Usenet (viz ICBM address). Pravděpodobně by zůstal za okrajem zájmu uživatelů, nebýt vzniku fenoménu jménem weblog. Weblogy jsou většinou úzce vázány na konkrétní osoby, které o sobě rády dávají vědět. Vznikla tedy iniciativa GeoURL, která registruje nejrůznější zdroje a poskytuje informace o jejich lokaci. Z historických důvodů se však hodnoty metaelementu "ICBM" uvádějí v desetinných hodnotách, které je zvláště pro naše kraje těžké zjisti - já jsem využil služeb serveru Maporama.com.

```
      <!-- GEOURL /-->
      <!-- Geographics location (altitude, latitude) - http://geourl.org /-->
      <!-- You can obtain exact location via http://www.maporama.com/share/ /-->
      <meta name='ICBM' content='0.0000, 0.0000' />
      <!-- GEOURL - end /-->
```

## Prohlížeče a jejich výmysly

Výrobci prohlížečů (browserů) od počátku vnášeli do (X)HTML různé prvky, kterými se jednak chtěli odlišit od konkurence a druhak jimi chtěli rozšířit možnosti designérů. Některé z těchto výmyslů ale zašly příliš daleko a obtěžují jak poskytovatele tak i spotřebitele obsahu. Naštěstí je možno tato "vylepšení" pomocí metaelementů deaktivovat. V následující skupině oblíbených metaelementů první zabrání zobrazování nástrojové lišty MSIE nad obrázky (vhodné u obrázků, které jsou jen grafickým doplňkem stránky), druhý donutí MSIE zobrazit standardní ovládací prvky (formuláře) bez ohledu na aktuální vzhled Windows a třetí znemožní MSIE vkládat a zpracovávat "inteligentní" značky ve stránce. Poslední metaelement pak donutí Operu, aby přestala automaticky přizpůsobovat rozměry obrázků velikosti viewportu, předstíraje tak MSIE.

```
      <!-- BROWSER SPECIFIC FEATURES = ALL OFF /-->
      <!-- MSIE - 'helpful' features /-->
      <meta http-equiv='imagetoolbar' content='no' />
      <meta http-equiv='MSThemeCompatible' content='no' />
      <meta name='MSSmartTagsPreventParsing' content='TRUE' />
      <!-- OPERA - image resizing /-->
      <meta name='autosize' content='off' />
      <!-- BROWSER SPECIFIC FEATURES = end /-->
```

## Ikona dokumentu či serveru

S značkou "link" měli tvůrci HTML původně velké plány, které však později degenerovaly do několika vcelku užitečných aplikací. Spíše pro pobavení lze využít "shortcut icon", který k dokumentu, potažmo celému serveru, připojuje ikonku, kterou pak prohlížeč zobrazí v adresním řádku, záhlaví lišty nebo v oblíbených složkách. Implementace je poměrně problematická, ani moderní prohlížeče nezacházejí s tímto prvkem zcela korektně. Nebudu zde rozebírat podrobnosti, krátký popis naleznete v článku Vlastní ikona stránek, v českém jazyce asi nejobjemnější je stať Ikona stránky, která obsahuje všechny podstatné informace o tomto tématu.

```
      <!-- ICON /-->
      <!-- Use icon for this page if you have one /-->
      <link rel='shortcut icon' type='image/x-icon' href='favicon.ico' />
      <!-- ICON end /-->
```

## Alternativní verze dokumentu

S rozvojem různých zařízení, pomocí nichž přistupujeme k internetu, se stává stále zajímavější možnost odkázat v hlavičce (X)HTML na jiný formát, v němž je dostupná stejná informace. Lze takto do dokumentu vložit odkaz na verzi pro WAP, PDA nebo na jednoduchou textovou verzi. V poslední době je obzvláště oblíbeným alternativním formátem RSS, které může nést prakticky jakékoli informace. Některé čtečky nebo roboty už dokáží přímo ze stránek vypreparovat odkaz na RSS kanál a poskytnout ho čtenáři, pokud je v hlavičce dokumentu uvedena deklarace, podobná následující:

```
      <!-- ALTERNATIVE VERSION - plain text, PDA version or RSS if you have one /-->
      <link rel='alternate' type='application/rss+xml' title='RSS Example.net'
href='http://www.example.net/rss.xml' />
      <!-- ALTERNATIVE VERSION - end /-->
```

## Navigace a vazby mezi dokumenty

Jedním z prvních úkolů elementu "link" bylo udržování logických vazeb mezi dokumenty a usnadnění pohybu v sadách souvisejících souborů. Na internetu je však většina dokumentů na sobě nezávislá, pohyb mezi nimi lépe zajišťují hypertextové odkazy a vyhledávání souvisejících informací zase umožňují specializované služby, takže se element "link" tímto způsobem rychle přestal používat a přežíval pouze v intranetech. Zdá se však, že by se mohl vrátit, díky dynamicky generovaným stránkám (odpadají problémy s konzistencí), které přitom kvůli SEO předstírají statičnost. Navíc jsou už k dispozici i prohlížeče, které umí vygenerovat z metadat navigační lištu, tvářící se jako jejich integrální součást - pro tradičně zaostávající MSIE existuje plug-in <LINK> Navigation Bar, navíc s celkem inteligentní podporou odkazů na alternativní verze dokumentu.

```
      <!-- NAVIGATION - based on logical relations of documents /-->
      <!-- homepage /-->
      <link rel='home' href='http://www.example.net' />
      <!-- searchpage /-->
```

```
      <link rel='search' href='http://www.example.net/search/' />
      <!-- title page of serial /-->
      <link rel='up' href='http://www.example.net/serial/this/' />
      <!-- first part of serial /-->
      <link rel='first' href='http://www.example.net/article/first/' />
      <!-- last part of serial /-->
      <link rel='last' href='http://www.example.net/article/last/' />
      <!-- previous part of serial /-->
      <link rel='prev' href='http://www.example.net/article/previous/' />
      <!-- next part of serial /-->
      <link rel='next' href='http://www.example.net/article/next/' />
      <!-- news, RSS /-->
      <link rel='newsfeed' href='http://www.example.net/rss.xml' />
      <!-- NAVIGATION - end /-->
```

V ukázce výše vidíte sekvenci elementů "link" různého typu (atribut "rel"), která společně vytváří základní navigační strukturu sady dokumentů. V praxi můžete jejich použití vidět přímo zde na Interval.cz, pokud si otevřete článek, který je součástí nějakého seriálu, například XHTML - element link (do budoucna je mým cílem umožnit návštěvníkům procházet touto metodou prakticky celý náš web). Opět se vyhnu nošení sov do Atén a odkážu vás na podrobnější článek Navigace pomocí elementu link.

## CSS neboli kaskádové styly

Poslední sekvencí, kterou můžete vidět na mém příkladu, je skupina pravidel a odkazů na externí soubory pravidel CSS, pomocí nichž je definován vzhled a částečně i chování (X)HTML dokumentu. Uvádím zde tento příklad pouze pro úplnost, neb předpokládám, že většina z vás je s CSS již dobře seznámena. Pokud vás zajímají podrobnosti, například o alternativních stylech nebo o speciálních stylech pro různá média, jsou vám k dispozici jiné články. Upozornil bych pouze na první část, kde je do kódu přímo vložena CSS definice vzhledu stránky jako takové - předchází se tak problémům s odlišným pojetím okrajů v různých prohlížečích.

```
      <!-- CASCADING STYLE SHEETS /-->
      <!-- INPAGE STYLE - pagemargin problem solving /-->
      <style type='text/css' media='all'>
         BODY, HTML {
            border: 0px none;
            margin: 0px;
            padding: 0px;
         }
      </style>
      <!-- LINKED STYLE /-->
      <link rel='stylesheet' type='text/css' media='all' href='./css/css_all.css' />
      <link rel='stylesheet' type='text/css' media='screen'
href='./css/css_screen.css' />
      <link rel='stylesheet' type='text/css' media='print' href='./css/css_print.css'
/>
      <!-- PREFFERED STYLE /-->
      <link rel='stylesheet' type='text/css' href='./css/css_pref.css'
title='Preffered style' />
      <!-- ALTERNATE STYLE /-->
      <link rel='alternate stylesheet' type='text/css' href='css_alt.css'
title='Alternate style' />
      <!-- CASCADING STYLE SHEETS - end /-->
```

# XHTML - základní struktura dokumentu

Vedle dodržení struktury XHTML dokumentu je vhodné nezapomínat na informace o jazyku dokumentu. Vysvětlím vám proč a současně vše doplním upřesněním elementu html.

## Základní struktura XHTML dokumentu

Každý XHTML dokument musí obsahovat tyto čtyři základní sekce, a to v uvedeném pořadí:

1. **XML deklarace** – tato sekce je povinná, jen pokud váš dokument používá jiné kódování než UTF-8 nebo UTF-16, ovšem silně se doporučuje uvádět ji pokaždé. Musí se nacházet na prvním řádku dokumentu.

2. **Deklarace typu dokumentu** – připomínám, že tato deklarace určuje, podle jakého DTD je dokument napsán. Pro XHTML 1.0 si můžete vybrat jednu z těchto deklarací:

**XHTML 1.0 Strict:**

```
<!DOCTYPE html


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

**XHTML 1.0 Transitional:**

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

**XHTML 1.0 Frameset:**

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

**Systémový identifikátor** (URI) v těchto deklaracích může být modifikován podle potřeb systému, aniž by se tím narušila valididita (správnost) dokumentu.

3. **Hlavička dokumentu** – tato sekce je celá uzavřena v elementu `head`. Uvádějí se v ní takzvaná metadata, tedy informace o dokumentu (například titulek dokumentu, stručné shrnutí jeho obsahu, jeho vztah k jiným dokumentům a podobně). Tyto informace prohlížeč nezobrazuje jako součást stránky, k jejich reprezentaci může (ale nemusí) použít jiné způsoby (například uvedení titulku dokumentu v titulku okna).

4. **Tělo dokumentu** – tato sekce se kompletně nachází uvnitř elementu `body`. Nachází se v ní samotný obsah stránky, který je určen k přímé interpretaci.

Hlavička a

tělo dokumentu (tedy elementy `head` a `body`) se navíc musí kompletně nacházet uvnitř **elementu** `html`. Element `html` musí obsahovat **deklaraci jmenného prostoru**, což vysvětlím za chvíli. Nyní se podívejte na příklad **minimálního XHTML dokumentu**, který by vyhověl specifikaci XHTML 1.0 Strict:

```
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <title>Minimální XHTML stránka</title>
  </head>
  <body>
    <p>Toto je minimální XHTML stránka, která by vyhověla specifikaci XHTML.</p>
```

```
  </body>
</html>
```

Tato stránka má několik „ale“. Jednak v ní chybí XML deklarace, takže musí používat kódování UTF-8 nebo UTF-16, neobsahuje deklaraci používaného jazyka (tedy češtiny) a také v ní chybí deklarace použitého kódování pro starší prohlížeče, které ho neumí určit z XML deklarace. Proto ji trochu poupravíme:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs" lang="cs">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
    <meta http-equiv="content-language" content="cs" />
    <title>XHTML stránka</title>
  </head>
  <body>
    <p>Toto již je XHTML stránka se všemi nutnými náležitostmi.</p>
  </body>
</html>
```

Významem elementů a atributů v příkladu si zatím nemusíte až tolik lámat hlavu, protože se k nim ještě dostaneme podrobněji. Přesto bych ale poznamenal, že u elementů `meta` udávající kódování a jazyk je vhodné, aby se nacházely v dokumentu dříve, než jakýkoli text v lidské řeči, aby interpreti XHTML ihned věděli, jak mají s tímto textem nakládat.

Nyní, když už víme, jak by měla vypadat základní XHTML stránka, můžeme přikročit k představení prvního elementu. Ještě předtím bych vás ale chtěl upozornit na soubor strict-dtd.html. Tento XHTML soubor obsahuje celé **XHTML 1.0 Strict DTD**, což znamená, že ho můžete otevřít v prohlížeči a pomocí odkazů se po něm snadno pohybovat (nevytvořil jsem ho já, jedná se o upravený soubor ze specifikace XHTML 1.0). To je důležité hlavně proto, že v celém DTD se velmi hojně užívá parametrických entit, které si nebudeme zdlouhavě vypisovat u každého elementu či atributu, a proto vás budu odkazovat na tento soubor, kde se pouhým kliknutím na jméno entity přenesete na její deklaraci. Elementy i atributy si budeme představovat pomocí fragmentů z DTD, kterými jsme se zabývali v úvodu seriálu. Pokud tedy něčemu nerozumíte, doporučuji vám předchozí části seriálu.

## Element html

**Povolený obsah:** (`head`, `body`)

**Atributy:**

**`%i18n;`**

Tato parametrická entita obsahuje atributy internacionalizace, tedy `lang`, `xml:lang` a `dir` – jejich významu a syntaxi se budeme věnovat dále v článku.

**`id  ID  #IMPLIED`**

Obsahem tohoto atributu je jméno elementu, které musí být jedinečné pro celý dokument. Tomuto atributu se budeme věnovat podrobně v příštím článku.

**`xmlns  %URI;  #FIXED 'http://www.w3.org/1999/xhtml'`**

Tento atribut deklaruje jmenný prostor XHTML, musí být přítomen v každém XHTML dokumentu a vždy musí mít hodnotu `http://www.w3.org/1999/xhtml`.

Element html v XHTML 1.0 Strict DTD

Element `html` je **kořenovým prvkem** XHTML dokumentů, tzn. musí v něm být obsažen celý obsah dokumentu. Povinně v něm musíte uvést atribut `xmlns` s danou hodnotou a měli byste uvést i atributy internacionalizace.

## Entita %i18n; – atributy internacionalizace

Atributy z této entity slouží k poskytnutí údajů o

přirozeném (lidském) **jazyku dokumentu**. Jedná se o tyto tři atributy:

```
lang    %LanguageCode;    #IMPLIED
xml:lang    %LanguageCode;    #IMPLIED
```

Tyto atributy shodně udávají **kód jazyka**, ve kterém je uveden celý jejich obsah včetně elementů (zde existují výjimky u tabulek, ke kterým se dostaneme v jednom z dalších dílů) i všechen text jejich atributů, pokud se jedná o text v přirozené lidské řeči. Pokaždé, když chcete definovat jazyk elementu, musíte k tomu v XHTML 1.0 použít jak atribut `lang`, tak i `xml:lang`, přičemž oba musí mít stejný obsah. První atribut kvůli zpětné kompatibilitě s HTML a prohlížeči HTML, ten druhý je pak speciálním atributem XML, vyhrazeným pro tento účel. V XHTML 1.1 se již používá pouze `xml:lang`.

Entitu `%LanguageCode;` jsme si představovali již dříve, proto nyní jen připomenu, že kódy jazyků jsou obsaženy v RFC 1766. Tyto kódy nejsou citlivé na velikost písmen. Za kódem jazyka můžete navíc použít řetězec `-varianta_jazyka`, například `en-US` (americká angličtina) nebo `x-klingon` (`x` udává experimentální označení).

```
dir    (ltr|rtl)    #IMPLIED
```

Tento atribut udává **směr textu** (`ltr` – *left-to-right* – pro text zleva doprava a `rtl` – *right-to-left* – pro text zprava doleva). Stejně jako element `bdo` najde využití pouze ve spojení s exotickými jazyky, a proto ho většina z nás nikdy nebude potřebovat. Z tohoto důvodu se těmito prvky XHTML nebudeme zabývat.

Nyní bych rád uvedl pár důvodů, proč je vhodné na informace o jazyku dokumentu nezapomínat:

- Vyhledávací roboti mohou vaši stránku snadno zařadit do jedné z jazykových skupin a uživatelé vás díky tomu snadno najdou.
- Hlasové prohlížeče mohou snadno určit, jakou výslovnost mají použít u vaší stránky.
- Vizuální prohlížeče mohou dodržovat typografická pravidla příslušející danému jazyku (což se týká zejména uvozovek a mezer).
- Automatické kontrolory pravopisu mohou s vaší stránkou lépe pracovat.

U variant jazyka chápou interpreti XHTML element jako psaný v základním jazyce, navíc určeném jeho zvláštní variantou. Vysvětlím to na příkladě: Pokud bychom měli dokument, který by měl v „jazykových atributech“ uvedeno `en-US` a hlasový prohlížeč obsahoval speciální výslovnost pro angličtinu, ale již ne pro americkou angličtinu, četl by dokument s anglickou výslovností. Pokud by ale obsahoval i výslovnost pro americkou angličtinu, četl by ho americkou angličtinou.

Díky dědičnosti stačí udat jazyk pouze pomocí základních mechanismů (element `meta` a atributy `lang`, `xml:lang` elementu `html`). Pokud potřebujeme použít uvnitř dokumentu jiný než takto nastavený jazyk, uděláme to nastavením „jazykových atributů“ u příslušného elementu (tyto atributy lze použít téměř u jakéhokoli elementu, u kterého to dává smysl). Vše si ukážeme na příkladě, ve kterém do české stránky vložíme odstavec v angličtině:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html
     PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
     "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="cs" lang="cs">
  <head>
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
    <meta http-equiv="content-language" content="cs" />
    <title>Použití angličtiny v české stránce</title>
  </head>
  <body>
    <p>odstavec česky <!--element p zde zdědil nastavení jazyka z elementu html --
></p>
    <p xml:lang="en" lang="en">a paragraph in <strong>English</strong> <!-- anglický
text, element p zde má své vlastní nastavení jazyka - to se aplikuje na jeho obsah
včetně vnořených elementů (zde je takovým elementem strong) i jeho atributy --></p>
  </body>
</html>
```

Tímto jsme tedy probrali atributy internacionalizace. Jak už jsem řekl, vyskytují se u většiny elementů, takže při jejich dalším probírání si už tuto entitu vysvětlovat nebudeme, pouze si připomeneme, že se jedná o atributy internacionalizace. Příště se budeme zabývat dalšími základními elementy a atributy.

# XHTML - povolené obsahy elementů

Již jsme se probrali v podstatě všemi elementy striktního XHTML. Poněkud jsme ale při tom zanedbávali obsah, který pro ně DTD povoluje. To v tomto článku napravíme.

Jak již z tohoto seriálu víte, povolený obsah elementů je reprezentován výčtem elementů (a

sekce #PCDATA) spojených určitými speciálními znaky. Mnoho elementů má ale totožný povolený obsah, proto autoři DTD XHTML vytvořili tři parametrické entity - `%Flow;`, `%Inline;` a `%Block;`.

Tyto parametrické entity reprezentují nejčastější povolené obsahy elementů a v průběhu seriálu jsme se s nimi mnohokrát setkali. Mnoho elementů má také z těchto parametrických entit svůj obsah odvozen - většinou je zde o nějaký ten element víc nebo míň.

Nyní si představíme tyto tři nejčastější entity - u každé z nich si zároveň řekneme, které elementy mají z této entity obsah odvozen. Ve všech případech se také dozvíte, pro jaké elementy daná parametrická entita definuje povolený obsah.

## Parametrická entita %Flow;

Parametrická entita `%Flow;` obsahuje text (#PCDATA) a tyto elementy:

- odstavec (`p`),
- nadpisy (`h1`, `h2`, `h3`, `h4`, `h5` a `h6`),
- kontejnery (`div` a `span`),
- seznamy

  a výčty (`ul`, `ol` a `dl`),

- blokové sémantické elementy (`blockquote` a `address`),
- externí objekty

  (`object` a `img`),

- řádkové sémantické elementy (`a`, `em`, `strong`, `dfn`, `code`, `q`, `samp`, `kbd`, `var`, `cite`, `abbr`, `acronym`, `sub` a `sup`),
- elementy formulářů (`form`, `fieldset`, `input`, `select`, `textarea`, `label` a `button`),
- a elementy `table`, `map`, `noscript`, `ins`, `del`, `script`, `pre`, `hr` a `br`.

Všechny tyto elementy a text se mohou vyskytovat v libovolném pořadí a množství.

Jak vidíte, do entity `%Flow;` se řadí téměř všechny elementy XHTML. Do elementů, které ji mají nastavenou jako svůj povolený obsah, proto můžete vnořit téměř všechno.

Entitu `%Flow;` mají nastavenou jako svůj povolený obsah elementy `div`, `li`, `dd`, `ins` a `del`.

### Obsahy elementů odvozené z parametrické entity %Flow;

Element `object` má jako svůj povolený obsah také nastavenu entitu `%Flow;`, navíc se zde ale ještě může vyskytovat element `param`.

Element `fieldset` také může obsahovat elementy z `%Flow;`, navíc se zde ale může vyskytovat element `legend` - ten se, pokud je přítomen, musí nacházet hned na začátku obsahu elementu, nesmí být předcházen žádným textem ani jinými elementy.

Element `button` smí obsahovat všechny elementy z `%Flow;` s výjimkou `a`, `form`, `input`, `select`, `textarea` a `button`.

## Parametrická entita %Inline;

Parametrická entita `%Inline;` obsahuje text (#PCDATA) a tyto elementy:

- řádkový kontejner `span`,
- řádkové sémantické elementy (`a`, `em`, `strong`, `dfn`, `code`, `q`, `samp`, `kbd`, `var`, `cite`, `abbr`, `acronym`, `sub` a `sup`),
- externí objekty (`object` a `img`),
- řádkové elementy formulářů (`input`, `select`, `textarea`, `label` a `button`),
- a elementy `map`, `ins`, `del`, `script` a `br`.

Všechny tyto elementy a text se mohou vyskytovat v libovolném pořadí a množství.

Entita `%Inline;` v sobě zahrnuje veškerý řádkový obsah - tedy text a řádkové elementy (či elementy, které mohou být blokové i řádkové). Elementy, které mají na tuto entitu svůj obsah nastaven, by měly obsahovat pouze text a s ním spojené elementy (viz výše) - nejsou určeny jako kontejnery pro další prvky.

Na `%Inline;` mají svůj obsah nastaven tyto elementy:

- odstavec (`p`),
- nadpisy (`h1`, `h2`, `h3`, `h4`, `h5` a `h6`),
- kontejner `span`,
- řádkové sémantické elementy s výjimkou `a` (`em`, `strong`, `dfn`, `code`, `q`, `samp`, `kbd`, `var`, `cite`, `abbr`, `acronym`, `sub` a `sup`),
- a elementy `dt`, `address`, `label`, `legend` a `caption`.

### Obsahy elementů odvozené z parametrické entity %Inline;

Element `pre` má nastaven svůj obsah na elementy z entity `%Inline;` s výjimkou elementů `img` a `object`.

Element `a` má také svůj obsah nastaven na `%Inline;`, výjimku zde ale tvoří element `a` (odkazy není možné vnořovat).

## Parametrická entita %Block;

Parametrická entita `%Block;` obsahuje pouze tyto elementy (ne text):

- odstavec (`p`),
- nadpisy (`h1`, `h2`, `h3`, `h4`, `h5` a `h6`),
- kontejner `div`,
- seznamy a výčty (`ul`, `ol` a `dl`),
- blokové sémantické elementy (`blockquote` a `address`),
- blokové formulářové elementy (`form` a `fieldset`),
- a elementy `table`, `noscript`, `ins`, `del`, `script`, `pre` a `hr`.

Všechny tyto elementy se mohou vyskytovat v libovolném pořadí a množství.

Entita `%Block;` v sobě sdružuje blokové elementy. Narozdíl od `%Flow;` a `%Inline;` ale neobsahuje text - ten do elementů, které na ni mají nastaven povolený obsah, přímo vkládat nemůžete.

Elementy, které mají nastaven povolený obsah na `%Block;`, jsou `noscript`, `body` a `blockquote`.

### Obsahy elementů odvozené z parametrické entity %Block;

Element `map` může obsahovat buď elementy z `%Block;` nebo elementy `area`.

Element `form` může také obsahovat elementy z `%Block;`, s jedinou výjimkou - element `form` (formuláře se nesmí vnořovat do sebe).

# XHTML - tělo dokumentu, kontejnery a odstavce

V dnešním článku se podíváme na element body, sloužící k uzavření obsahu stránky, kontejnery div a span, jež nám umožňují dělit obsah na specifické oblasti, a element p, který vytváří odstavce.

## Element body – tělo dokumentu

**Povolený obsah:** `%Block;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`onload %Script; #IMPLIED`

Tato událost se aktivuje ve chvíli, kdy je celá stránka nahrána do prohlížeče. Lze ji použít pouze u elementů `body` a `frameset` (tím se budeme zabývat v části o rámcích).

`onunload %Script; #IMPLIED`

Tato událost se aktivuje ve chvíli, kdy je stránka odstraněna z okna prohlížeče, např. zavřením okna nebo přechodem na jinou stránku. Stejně jako událost `onload` ji lze použít pouze u elementů `body` a `frameset`.

Element body v XHTML 1.0 Strict DTD

Element `body` uzavírá celý **obsah stránky**. Ten může být reprezentován různými způsoby – vizuálně na obrazovce počítače, hlasově pomocí čtečky obrazovky nebo může být třeba vytištěn na braillovské tiskárně.

Element `body` je **povinný** pro každý XHTML dokument. Jeho obsahem mohou být elementy z parametrické entity `%Block;`, kterou si představíme později. Prozatím nám bude stačit, že nemůžeme do `body` vložit přímo text, ale musíme ho obalit ještě do nějakého jiného elementu.

Také bych vás rád upozornil na zajímavou možnost, a

sice jednoznačně pojmenovat element `body`:

```
<body id="www-interval-cz">
```

Možná se teď ptáte, k čemu je to dobré, vždyť element `body` se může na stránce vyskytovat jen jednou a není ho proto třeba pojmenovávat. Je to kvůli uživatelským stylům (to jsou styly, které definuje uživatel a prohlížeč je potom aplikuje na každou navštěvovanou stránku). Tím, že svoje stránky takto pojmenujete, umožníte uživateli, aby do svých uživatelských stylů zařadil speciální pravidla pouze pro vaši stránku, a mohl si ji tak maximálně přizpůsobit svým potřebám. Tento jev se nazývá **CSS signatures** a zatím se nejedná o žádný standard, spíše nápad. Pokud se jej rozhodnete využívat, obsahem vašeho `id` by měla být adresa stránky, kde převedete tečky na spojovníky (-).

## Elementy div a span – kontejnery

### Element div

**Povolený obsah:** `%Flow;`

**Atributy:** `%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element div v XHTML 1.0 Strict DTD

### Element span

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element span v XHTML 1.0 Strict DTD

Elementy `div` a `span` slouží jako kontejnery – můžete s nimi označit nějaký blok obsahu a přiřadit mu určité vlastnosti, například pomocí CSS a skriptů (v tomto případě pravděpodobně využijete atributy `class` a `id` k identifikaci bloku), nebo nastavit pro tento blok odlišný jazyk než v kterém je psán dokument.

Oba elementy nemají **žádný sémantický význam**. To znamená, že jejich použitím neříkáte nic o významu obsahu, který je do nich uzavřen. Slouží pouze jako kontejnery.

Oba elementy se od sebe liší vcelku zásadně. Element `div` je blokový element a jeho povolený obsah `%Flow;` nám říká, že do něj můžeme vkládat text a další blokové i řádkové elementy v jakémkoliv pořadí (k parametrické entitě `%Flow;` se ještě dostaneme). Naproti tomu `span` je řádkový element a může obsahovat pouze řádkové elementy a text (to nám říká parametrická entita `%Inline;`, kterou se také ještě budeme zabývat). Příklad použití:

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE html

     PUBLIC '-//W3C//DTD XHTML 1.0 Strict//EN'
     'http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd'>
<html xmlns='http://www.w3.org/1999/xhtml' xml:lang='cs' lang='cs'>
<head>
  <meta http-equiv='content-type' content='text/html; charset=UTF-8' />
  <meta http-equiv='content-language' content='cs' />
  <title>XHTML stránka</title>
</head>
<body>

<div id='page'>
<p>Tady je český text. Nyní budeme přidávat trochu anglického textu: <span
xml:lang='en' lang='en'>Here isthe English text.</span></p>
</div>

</body>
</html>
```

## Element p – odstavec

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element p v XHTML 1.0 Strict DTD

Element `p` slouží k vytvoření odstavce. Může obsahovat pouze text a řádkové elementy, nelze do něj uzavřít například tabulku, formulář, element `div`, nadpis, nebo další element `p`. Ke kompletnímu popisu entity `%Inline;`, která reprezentuje jeho obsah, se ještě v seriálu dostaneme.

Ačkoliv to DTD nezakazuje, neměli byste používat odstavce bez jakéhokoliv obsahu (`<p></p>`). Tyto odstavce by měly být interprety XHTML ignorovány.

Odstavce jsou v drtivé většině vizuálních prohlížečů formátovány jako **samostatné bloky s vertikálními okraji** zarovnané vlevo. Například tento text...

```
<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis volutpat. Duis leo.
Donec laoreet iaculis quam. Vivamus ipsum. Vestibulum feugiat, neque pretium aliquet
pulvinar, nunc diam fringilla mi, sit amet pulvinar nibh enim vel ipsum.</p>
<p>Nulla a nibh nec turpis ornare tempus. In commodo. Morbi eu velit. Vivamus
euismod, tortor nec varius lobortis, purus tellus vestibulum libero, ut sagittis
mauris sapien in tellus. Integer nec eros eu quam fringilla rhoncus. Vestibulum
pulvinar.</p>
```

...bude formátován asi takto:

```
Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis volutpat. Duis leo.
Donec laoreet iaculis quam. Vivamus ipsum. Vestibulum feugiat, neque pretium aliquet
pulvinar, nunc diam fringilla mi, sit amet pulvinar nibh enim vel ipsum.

Nulla a nibh nec turpis ornare tempus. In commodo. Morbi eu velit. Vivamus euismod,
tortor nec varius lobortis, purus tellus vestibulum libero, ut sagittis mauris sapien
in tellus. Integer nec eros eu quam fringilla rhoncus. Vestibulum pulvinar.
```

Zobrazení odstavců můžete změnit pomocí stylů, ale nedoporučuji vám formátovat odstavce bez zmiňovaných vertikálních okrajů. Weboví uživatelé neradi čtou dlouhé bloky textu (a odstavce bez okrajů vypadají jako dlouhý blok textu) a raději je přeskakují. Také byste měli dodržovat pravidlo „jedna myšlenka = jeden odstavec“ – text se na obrazovce čte hůře než na papíru, proto bychom to měli uživatelům co nejvíce ulehčit jeho **vhodným strukturováním**.

# XHTML - odkazy

Dnes si budeme povídat o odkazech – řekneme si, jak je vytvářet, ale především jak je správně používat. Tento článek ve skutečnosti předchází minulému článku, při jeho korekturách však bylo jejich vzájemné pořadí přehozeno, za což se vám redakce Intervalu omlouvá.

## Co jsou to odkazy?

Odkaz (nebo také *hyperlink* či zkráceně *link*) je obecně **místo v XHTML dokumentu, pomocí kterého může uživatel přejít na jiné místo**, které se nachází buď v aktuálním dokumentu nebo kdekoli jinde na internetu (v takovém případě může jít například o jiný XHTML dokument, obrázek, video...). K přemístění na nové místo dojde po aktivaci odkazu, například kliknutím myší, stiskem klávesy nebo hlasovým příkazem.

Jako uživatelům webu jsou vám odkazy určitě důvěrně známé. Jedná se o jeden ze základních kamenů webu, je to jakási specialita webových dokumentů – to, co dělá z jednotlivých dokumentů umístěných ve všech koncích světa jednu velkou „pavučinu“.

## Jakou funkci mají odkazy?

- Uživatel se pomocí nich může pohodlně pohybovat po webu.
- Odkazy slouží ke zvýraznění textu. Weboví uživatelé, známí svou nechutí číst dlouhé texty, často skáčou očima po jednotlivých odkazech. Pokud jsou texty odkazů zajímavé a jednoduché, potom přikročí k detailnějšímu čtení stránky. Pokud jsou ale nicneříkající (např. „zde“, „dále“ apod.), potom je stránka přestává zajímat.
- Odkazy specifickým způsobem obohacují styl psaní na webu. Představte si, jak by vypadala předcházející věta, kdyby v ní chyběl odkaz a stála sama o sobě. Ano, hloupě, protože nijak neobhajuje tvrzení, které je jejím obsahem. Pokud ji ale uvedeme s odkazem, hned je vidět, že za tvrzením stojí ještě nějaké argumenty – dokonce potom věta vypadá tajemně, vybízí uživatele podívat se, co se skrývá za daným odkazem.
- Pomocí odkazů se po webu pohybují roboti vyhledávačů, a proto je třeba volit je citlivě a konzistentně.
- Zdroje, které uvádíte pomocí odkazů, jsou součástí dojmu

  , který budou mít uživatelé z vašeho webu.

Samozřejmě, že jsem neuvedl všechny vlivy, které mají odkazy na vaší stránku. Chtěl jsem pouze ilustrovat, o jak důležitou součást stránky se jedná. Odkazům byste měli věnovat tu největší námahu, protože to za to jednoduše stojí a vydané úsilí se vám nepochybně vrátí.

## Element a – vytváření odkazů

**Povolený obsah:** `%a.content;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`%focus;`

Tato parametrická entita obsahuje atributy specifické pro elementy, které mohou obdržet tzv. *fokus* – ten se uplatňuje obvykle při procházení stránky klávesnicí (typicky pomocí klávesy `Tab`). Řadíme sem atributy `accesskey`, `tabindex`, `onfocus` a `onblur`. Kromě odkazů mohou fokus obdržet také klikací mapy (forma odkazů pro efektivní používání obrázků) a prvky formulářů (k ouběma tématům se v seriálu ještě dostaneme). Touto entitou se budeme zabývat v příštím díle.

```
href        %URI;        #IMPLIED
```

Tento atribut specifikuje URI adresu odkazovaného zdroje. Můžeme zde použít i tzv. identifikátor fragmetu, pomocí kterého se odkážeme na určité místo v (X)HTML dokumentu – tomuto tématu se věnujeme níže.

```
charset        %Charset;        #IMPLIED
```

Tento atribut specifikuje kódování zdroje, na který je odkazováno. Uvádí se obvykle pouze pokud je zdrojem (X)HTML dokument, ale jeho použití není nutné a leckdy ani možné.

```
type        %ContentType;        #IMPLIED
```

Pomocí tohoto atributu říkáte, jakého MIME-typu je zdroj, na který odkazujete. Interpret XHTML se pak může rozhodnout, jestli bude zdroj přenášet nebo ne (pokud tento MIME-typ nepodporuje).

```
name        NMTOKEN        #IMPLIED
```

Tento atribut pojmenovává daný odkaz, aby se na něj mohl odkazovat jiný odkaz. V XHTML 1.0 se nachází pouze kvůli kompatibilitě se staršími prohlížeči, ale v XHTML 1.1 se již nevyskytuje. Místo něj byste měli používat atribut `id`. Odkazováním se na konkrétní místa v dokumentu se zabýváme dále v článku.

```
hreflang        %LanguageCode;        #IMPLIED
```

Tento atribut určuje základní jazyk odkazovaného zdroje.

```
rel        %LinkTypes;        #IMPLIED
```

Tento atribut specifikuje vztah odkazovaného zdroje k aktuálnímu dokumentu. Např. pokud máme soubor `druha-kapitola.html`, který je druhou částí v nějaké souvislé sérii dokumentů, potom bychom odkaz z tohoto souboru na soubor `treti-kapitola.html`, který je třetí částí v té samé sérii, mohli doplnit o atribut `rel="next"` (tedy další dokument v sérii). Možným hodnotám tohoto atributu se budeme věnovat v příštím díle.

```
rev        %LinkTypes;        #IMPLIED
```

Tento atribut je podobný jako atribut `rel`, ovšem specifikuje vztah aktuálního dokumentu k odkazovanému dokumentu. Pokud bychom uvážili předchozí příklad, potom bychom mohli odkaz z druhého dokumentu na třetí dokument doplnit o atribut `rev="prev"` (tedy předchozí dokument z dané série), protože druhý dokument je předchozím dokumentem pro třetí dokument. Opět opakuji, že možným hodnotám tohoto atributu se budeme věnovat v příštím díle.

```
shape        %Shape;        "rect"
```

Tento atribut se používá, pokud je odkaz používán jako obrázková mapa – proto se mu budeme věnovat až v díle o tomto tématu.

```
coords        %Coords;
#IMPLIED
```

Stejně jako předchozí atribut i tento se používá, pokud je odkaz používán jako obrázková mapa a stejně tak se mu budeme věnovat až v díle o tomto tématu.

Element a v XHTML 1.0 Strict DTD

**Obsahem** elementu `a` (z angl. *anchor*, tedy *kotva*) je text nebo externí objekt (například obrázek), který se má stát obsahem odkazu. Tento obsah je v DTD deklarován na speciální parametrickou entitu `%a.content;` – není na ní ale nic zvláštního, jedná se pouze o entitu `%Inline;` bez elementu `a`, protože není povoleno odkazy vnořovat do sebe. Parametrickou entitu `%Inline;` jsme si ještě nepředstavovali, ale již jsme se se ní setkali u elementů `p` a `span`, může obsahovat nejrůznější řádkové elementy. Sám může být element `a` obsažen pouze v těle dokumentu (ne ale přímo v elementu `body`):

```
...
<body>
<p><a href="http://interval.cz">Interval.cz – web, grafika a e-
komerce</a></p>
</body>
...
```

Tento odkaz se zobrazí asi takto: `Interval.cz – web, grafika a e-komerce`

Další příklad, tentokrát s využitím několika atributů elementu `a`:

```
...
<html xmlns="http://www.w3c.org/1999/xhtml" lang="cs" xml:lang="cs">
...
<body>
...
<a href="http://the-best-web-on-the-web.com" hreflang="en" charset="utf-8"
type="application/xhtml+xml"
   lang="en" xml:lang="en" title="The Best Web on the Web">an english
page</a>
...
</body>
</html>
```

Každý odkaz by měl sám o sobě, bez jakéhokoli kotextu, **popisovat zdroj**, na který odkazuje, mělo by z něj být jasně patrné, kam vede. Je to kvůli tomu, že odkazy jsou často procházeny bez kontextu. Například hlasové prohlížeče často nejdříve předloží uživateli seznam

odkazů a až poté případně prochází celou stránkou, opomenout nemůžeme ani vyhledávače. Není kvůli tomu naštěstí nutné dávat celý popis jako text odkazu, k úplnému popisu můžeme využít atribut `title`. Příklad:

```
Tvorbou webu se zabývá mnoho serverů, ale jenom málo z nich <a
href="http://interval.cz" title="Interval.cz – web, grafika a e-
komerce">profesionálně</a>.
```

Ve vizuálních prohlížečích bývají odkazy podtržené a navíc odlišené barvou. Toto zobrazení můžete změnit pomocí stylů, ale rozhodně by i potom měly být odkazy dostatečně kontrastní vzhledem k normálnímu textu.

Atributy `name` a `href` můžete použít současně v jednom elementu `a`, což znamená, že jeden element `a` může být zároveň odkazem i cílem jiného odkazu. Pokud ale element `a` slouží pouze jako cíl jiného odkazu, potom byste zde neměli používat žádné z atributů pro bližší určení zdroje: `charset`, `type`, `hreflang`, `rel` a `rev`.

U **atributů pro bližší určení zdroje** ještě chvíli zůstaneme. Pokud je použijete, bude se vám zdát, že nemají v podstatě žádný praktický význam, vizuální prohlížeč interpretuje odkaz stejně, jako by tam nebyly. Nenechte se ale mýlit. Jejich důsledným používáním se zlepší logická struktura vašeho webu, což mimo jiné kladně ocení vyhledávací roboti. Navíc rozličná malá zařízení pro přístup na internet, která se již dnes prosazují, nemají tak skvělé výpočetní možnosti jako dnešní počítače, a proto nemohou, na rozdíl od dnešních počítačů, zobrazit každý soubor, na který odkazujete. Podobně jsou na tom i zařízení pro handicapované (i když zde se jedná spíše o to, co je schopen přijímat jejich uživatel). Uváděním těchto atributů jejich uživatelům velmi usnadníte práci. Posledním a možná nejpádnějším důvodem pro uvádění těchto atributů je možnost odlišit různé typy odkazů ve stylech (například pomocí atributových selektorů CSS).

### Odkaz na konkrétní místo v dokumentu

Pokud si v dokumentu pojmenujeme určité místo, můžeme se na něj odkazovat tak, že v URI adrese za jménem dokumentu uvedeme znak „#“ (zvaný též "sharp", "hash" a pod.), následovaný jménem, které jsme místu přidělili (v tomto jménu bychom měli dodržet přesně velká a malá písmena). Pokud tedy máme soubor `dokument1.html` na serveru `http://mujserver.cz`, v kterém se nachází pojmenované místo `tady`, potom bychom mohli na toto místo vytvořit odkaz následujícím způsobem:

```
<a href="http://mujserver.cz/dokument1.html#tady">text odkazu</a>
```

Samozřejmě, pokud se dokument nachází na stejném serveru, můžeme použít i relativní URI. Pokud se chceme odkázat na nějaké místo v aktuálním dokumentu, stačí pouze uvést „#“ následované jménem místa:

```
...
<body>
<p><a name="prvni_odstavec" id="prvni_odstavec">ahoj!</a></p>
<p><a href="#prvni_odstavec">první odstavec</a></p>
```

```
</body>
...
```

Již jsme to nakousli, tak tedy jak vytvořit místo v XHTML dokumentu, na které se lze odkazovat? Musíme pojmenovat určitý fragment kódu, což se v XHTML dělá pomocí atributu `id`, který můžeme použít prakticky u všech elementů. Tedy například takto:

```
<div id="obsah">
...
</div>
```

Takovéto odkazování je zcela v souladu s nejnovějšími standardy a ovládají ho všechny novější prohlížeče. Starší prohlížeče ovšem rozpoznají místo, na které se lze odkazovat, pouze pokud je tvořeno pomocí elementu `a` a jeho atributu `name`. Proto je v XHTML 1.0 tento atribut u elementu `a` zachován, ovšem je již zavržený a nedoporučovaný. Vždy, když ho u elementu `a` použijete, měli byste pužít i atribut `id` se stejnou hodnotou

(jako jsme to udělali v příkladu výše). Také byste se měli vyhnout elementům `a` bez jakéhokoli obsahu – nejlépe je do něj uzavřít nějaký text (nemusíte se bát, že by ho vizuální prohlížeče v takovém případě interpretovaly jako běžný odkaz, protože zde není přítomen atribut `href`).

# XHTML - nadpisy, informace o autorovi a změny v dokumentu

Podívejme se podrobněji na problematiku rozčlenění dokumentu na logické úseky pomocí nadpisů, poskytování informací o autorovi a vyznačování změn v dokumentu.

## Elementy h1, h2, h3, h4, h5 a h6 – nadpisy

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element h1 v XHTML 1.0 Strict DTD

Element h2 v XHTML 1.0 Strict DTD

Element h3 v XHTML 1.0 Strict DTD

Element h4 v XHTML 1.0 Strict DTD

Element h5 v XHTML 1.0 Strict DTD

Element h6 v XHTML 1.0 Strict DTD

Elementy nadpisů stručně popisují část dokumentu, kterou předcházejí. Pomocí nich byste měli dokument **rozčlenit** na strukturální bloky. Interpret XHTML je také může využít k vytvoření obsahu daného dokumentu.

Jak vidíte, existuje **šest úrovní** nadpisů – od `h1`, který má nejvyšší důležitost a v každém dokumentu by se měl vyskytovat právě jednou, po `h6` s nejmenší důležitostí. V dokumentu byste navíc neměli přeskakovat jednotlivé úrovně nadpisů, tzn. nepoužívat `h3` hned po `h1` apod.

**Vizuální prohlížeče** obvykle vykreslují nadpisy jako blokové elementy tučným zvětšeným písmem (velikost logicky odpovídá důležitosti nadpisu, takže např. `h1` je největší), ale toto zobrazení můžete změnit pomocí stylů.

Nyní příklad:

```
<h2>Renesance<h2>
<p>Mezi renesanční autory řadíme například Giovanniho Boccaccia, Francesca
Petrarcu nebo Williama Shakespearea.<p>
  <h3>Giovanni Boccaccio</h3>
  <p>Tento italský spisovatel se proslavil hlavně svou novelou Dekameron.</p>
    <h4>Dekameron</h4>
    <p>Toto dílo začíná ve Florencii v době moru. Sedm paní a
tři pánové odtud utíkají na venkov, kde zůstanou po deset dní, během kterých si
vyprávějí nejrůznější příběhy.</p>
  <h3>Francesco Petrarca</h3>
  <p>Další italský autor, který se proslavil hlavně svými sonety.</p>
...
```

## Element address – informace o autorovi

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element address v XHTML 1.0 Strict DTD

Tento element se používá k poskytování informací o

autorovi dokumentu nebo nějaké jeho větší části. Většinou se vyskytuje na začátku nebo na konci dokumentu. Např.:

```
<address>
<a href="mailto:martin@snizekweb.cz" title="e-mail">Martin Snížek</a>,
5. února 2003
</address>
```

Element `address` je blokový a stejně jako nadpisy nebo odstavce může obsahovat pouze text a řádkové elementy (kvůli parametrické entitě `%Inline;`, na kterou má deklarován svůj obsah).

## Elementy ins a del – označování změn v dokumentu

**Povolený obsah:** `%Flow;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

| | | |
|---|---|---|
| **cite** | **%URI;** | **#IMPLIED** |

Tento atribut má obsahovat adresu dokumentu, který popisuje důvod změny v dokumentu. Tento důvod může být popsán i v atributu `title`.

| | | |
|---|---|---|
| **datetime** | **%Datetime;** | **#IMPLIED** |

Tento atribut specifikuje datum

a čas změny ve formátu ISO:
YYYY-MM-DDThh:mm:ssTZD

YYYY - reprezentuje rok (čtyři čísla)
MM – reprezentuje měsíc (01-12)
DD – reprezentuje den v měsíci (01-31)
hh – reprezentuje hodinu (00-23)
mm – reprezentuje minutu (00-59)
ss – reprezentuje vteřinu (00-59)
TZD – reprezentuje časové pásmo: Z znamená UTC (Coordinated Universal Time), jinak musíme použít zápis +hh:mm nebo −hh:mm, podle toho, jestli je náš čas před UTC (+) nebo za UTC (−) – hh znamená počet hodin před/za UTC, mm počet minut

Pokud není znám přesný počet hodin, minut nebo vteřin, může na jejich místě být použit zápis 00
.

Tento typ je citlivý na velikost písmen, tzn. „T“ a „Z“ musí být uvedena velká.

Element ins v XHTML 1.0 Strict DTD

Element del v XHTML 1.0 Strict DTD

Uvedené elementy se používají k vyznačení částí dokumentu, které byly vloženy (`ins`) nebo odstraněny a déle již neplatí (`del`), a to pokud chcete tento stav u nich zdůraznit. Příklad:

```
<p>Příští setkání se koná <del cite="proc.html" datetime="2003-04-
10T00:00:00+1:00">12. 4.</del><ins>15. 4.</ins>.</p>
```

Z hlediska XHTML je na těchto elementech zajímavé, že se mohou chovat **jako blokové i řádkové**, ale ne současně. Tzn. pokud vystupují jako řádkové (tak je to i v našem příkladě, protože element `p` může obsahovat pouze řádkové elementy), nemohou obsahovat blokové elementy, jako jsou odstavce nebo nadpisy. Také mohou vystupovat jako blokové:

```
<del>
<p>Příští číslo vyjde: 18. 5. 2003</p>
</del>

<ins>
<p>Příští číslo již nevyjde.</p>
</ins>
```

Interpretace těchto elementů je různá, může být použito např. přeškrtnuté písmo pro `del`, odlišný font pro `ins` apod., toto zobrazení můžete změnit pomocí stylů.

# XHTML - hlavní sémantické elementy

Hlavní sémantické elementy XHTML jsou takové elementy, které říkají něco o obsahu, který obklopují, a umožňují tak i počítačům porozumět textu alespoň na minimální úrovni.

## „Phrase“ elementy – em, strong, dfn, code, samp, kbd, var, cite, abbr a acronym

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element em v XHTML 1.0 Strict DTD

Element strong v XHTML 1.0 Strict DTD

Element dfn v XHTML 1.0 Strict DTD

Element code v XHTML 1.0 Strict DTD

Element samp v XHTML 1.0 Strict DTD

Element kbd v XHTML 1.0 Strict DTD

Element var v XHTML 1.0 Strict DTD

Element cite v XHTML 1.0 Strict DTD

Element abbr v XHTML 1.0 Strict DTD

Element acronym v XHTML 1.0 Strict DTD

Pomocí tzv. *phrase* elementů vyznačujeme **význam** textu. Všechny zmiňované elementy jsou řádkové a mohou obsahovat pouze text nebo další řádkové elementy. Sémantika jednotlivých elementů je následující:

**`em`**
Vyznačuje zvýraznění (*emphasis*).
**`strong`**
Označuje důraznější zvýraznení.
**`dfn`**
Obsahem tohoto elementu je pojem nebo definice (*definition*).
**`code`**
Používá se k označení počítačového nebo programového kódu.
**`samp`**
Vyznačuje vzorový výstup programů, skriptů apod.
**`kbd`**
Indikuje text, který má být zadán uživatelem.
**`var`**
Označuje proměnnou, její vzorovou hodnotu
apod.
**`cite`**
Tento element označuje citovaný zdroj, odkaz na další zdroje nebo citaci. Poměrně se vžil zvyk uzavírat do něj jména osob, organizací apod.
**`abbr`**

Označuje zkratku (*abbreviation*), jejíž plné znění by se alespoň při prvním výskytu v dokumentu mělo nacházet v jeho atributu `title`. Příklady zkratek jsou XHTML, URI, ČR, nebo třeba ÚV KSČ ;-)

**`acronym`**

Tento element se používá k uzavírání zkratkových slov. Ty se narozdíl od zkratek vyslovují většinou jako jedno slovo, ne po jednotlivých písmenech. Pravidla pro použití atributu `title` jsou zde stejná jako u elementu `abbr`. Příklady zkratkových slov jsou NATO, NASA nebo Čedok.

Všechny tyto elementy lze používat poměrně **volně**, tzn. že můžete jejich význam různě odvozovat; nic není dáno ve specifikaci úplně na pevno. Např. zde na Intervalu se používá element `kbd` k uzavírání klávesových zkratek programů a různých cest v menu, pokud se o nich autor v článku zmiňuje.

Co se týče **zobrazení** phrase elementů ve vizuálních prohlížečích – `em`, `dfn`, `var` a `cite` jsou obvykle vykresleny kurzívou; `strong` tučným písmem a

`code`, `samp` a `kbd` písmem s pevnou šířkou znaků. Toto zobrazení můžete samozřejmě změnit pomocí stylů tak, aby odpovídalo celkovému pojetí vašeho serveru a informaci, kterou tyto elementy na vašem webu uzavírají. Elementy `abbr` a `acronym` nebývají formátovány nijak, proto je u nich použití stylů téměř nutností, aby uživatel vůbec mohl vidět, že se pod zkratkou skrývá její popis. V tomto případě se používá obvykle matné nesouvislé podtržení a kurzor myši indikující nápovědu.

Kromě vizuálních prohlížečů jsou ale tyto elementy použitelné i v jiných výstupech – např. textový prohlížeč, čtečka obrazovky... Jejich interpretace zde již většinou záleží na preferencích uživatele.

## Elementy q a blockquote – citace

**Povolený obsah:** (`q`) `%Inline;`

**Povolený obsah:** (`blockquote`) `%Block;`

**Atributy:**

**`%attrs;`**

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

**`cite`** **`%URI;`** **`#IMPLIED`**

Tento atribut udává adresu dokumentu, z kterého citace pochází.

Element q v XHTML 1.0 Strict DTD

Element blockquote v XHTML 1.0 Strict DTD

Tyto dva elementy slouží k uzavírání **citací**. Element `q` je řádkový a může obsahovat pouze text a řádkové elementy, proto se hodí pro kratší citace uvnitř textu. Naproti tomu element `blockquote`, jak už název napovídá, je blokový a může přímo obsahovat pouze blokové elementy, ne tedy hned text – ten musí být uzavřen v nějakém „kontejneru“, nejčastěji v odstavci.

Element `q` má ještě jednu zvláštnost. Kolem textu v něm uzavřeném by se ve vizuálním prohlížeči měly automaticky doplnit uvozovky podle jazyka, v kterém je dokument psán, neměli byste je tedy vkládat do textu. Tyto uvozovky by v případě potřeby měly být i zanořeny (pokud se citace nachází uvnitř jiné citace). Přesné zobrazení uvozovek můžete nastavit pomocí stylů. U elementu `blockquote` se automatické uvozovky nevyskytují.

`q` nemá obvykle žádné implicitní vizuální formátování, naopak obsah `blockquote` je většinou odsazen z obou stran. Toto zobrazení můžete samozřejmě přepsat ve stylech.

## Proč používat sémantické elementy?

A proč je tak důležité zabývat se sémantickými a strukturálními elementy, jako jsou dnes probírané *phrase* elementy, nadpisy, tabulky (těmi se budeme zabývat později) apod.? Pokud s nimi svůj text pečlivě a bohatě vyznačujete, dostáváte do rukou poměrně velké možnosti:

- Pomocí stylů můžete jednoduše definovat konzistentní zobrazení na všech stránkách vašeho webu

  . Můžete vizuálně odlišit různé úseky textu a pomoci tak uživateli v orientaci.

- Vyhledávače mají rádi dobře značkovaný text, protože jim pomáhá vaší stránce porozumět a předkládat tak uživateli lepší výsledky, proto se vám také odvděčí vyšší pozicí ve výsledcích vyhledávání.
- Správně strukturovaný text může být interpretován podle možností daného zařízení a preferencí uživatele. Hlavně u různých „alternativních“ zařízení může uživateli v mnohém ulehčit práci s vaší stránkou.
- Strukturovaný text vám otevírá možnosti strojového zpracování. Můžete např. jednoduše zrealizovat vyhledávání určitých pojmů (které ve všech textech vyznačujete stejným způsobem) jako jsou jména osob (pokud je uzavíráte např. do `cite`), termíny z dané oblasti (uzavřené např. v elementu `dfn`) apod. Také můžete vaše texty snadno analyzovat, např. spočítat počet zmínek o nejrůznějších pojmech... To jde samozřejmě zrealizovat i bez zmiňovaného značkování, ovšem výsledky nikdy nebudou stoprocentní, protože stroj jednoduše neví, jak jste kdy které slovo mysleli.

## Příklady

Na závěr několik příkladů na použití elementů, které jsme si dnes představili:

```
<dfn>Plynové trouby</dfn> se obvykle používají k pečení.

Do následujícího pole vložte <kbd>jméno</kbd>.

Nadpisy v XHTML jsou velmi důležité pro <strong>vyhledávače</strong>.

Webové stránky se posílají pomocí protokolu <abbr title="HyperText Transfer
Protocol">HTTP</abbr>.

Odstavec se v XHTML vytváří pomocí elementu <code>p</code>.

<q>Jediné vítězství nad láskou je útěk</q>, řekl kdysi <cite>Napoleon
Bonaparte</cite>.

<blockquote>
<p>Dobrý manžel má hodnotu dvou dobrých manželek, neboť věci mají tím větší hodnotu,
čím jsou vzácnější.</p>
</blockquote>
```

# XHTML - další textové elementy

V dnešním článku se podíváme, jak vytvořit takzvaný subskript a superskript, budeme se zabývat zalamováním řádků, speciálními znakovými entitami a také si představíme zvláštní element pre.

## Elementy sub a sup - subskript a superskript

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element sub v XHTML 1.0 Strict DTD

Element sup v XHTML 1.0 Strict DTD

Elementy `sub` a `sup` se využívají k vytvoření tzv. subskriptu, ($_{takto}$) respektive superskriptu ($^{takto}$). Měli byste je ale používat jen tehdy, je-li sub/superskript pro text v nich uzavřený důležitý k přenesení významu (například $H_2O$ nebo 2

[2]), ne k dekoračním účelům (k tomu slouží styly)!

### Element br - zalomení řádku

**Povolený obsah:** `EMPTY` (*žádný obsah*)

**Atributy:**

`coreattrs;`

Do této parametrické entity řadíme atributy `id`, `class`, `style` a `title`.

Element br v XHTML 1.0 Strict DTD

Element `br` v místě svého výskytu ukončí řádek, takže další text začíná až na následujícím řádku.

## Automatické zalamování řádek

Vizuální prohlížeč má k dispozici na šířku pouze omezenou plochu, proto musí logicky u všech delších textů docházet k jejich zalamování. Algoritmus zalamování záleží na prohlížeči, ale žádný doposud neovládá automatické dělení slov. Poslední slovo (zde je chápáno slovo jako sekvence znaků, jež nejsou bílými místy), které se nevejde na řádek, se pak většinou přesune na další.

K **rozdělení** pak ale může dojít tam, kde k němu podle typografických či jiných pravidel docházet nemá. Naštěstí obsahuje XHTML tzv. nedělitelnou mezeru ` `, o které jsme se již jednou zmiňovali v části o

pravidlech chování interpretů XHTML. Kamkoli tuto entitu vložíte, vykreslí se normální mezera, ale již nemůže dojít k zalomení mezi danými dvěma slovy.

XHTML navíc obsahuje ještě jeden speciální znak, a sice tzv. **měkký spojovník**, reprezentovaný entitou `&shy;` (*soft hyphen*). Pokud ho vložíte mezi písmena slova, rozdělí si podle něj interpret XHTML při zalamování slovo na části, se kterými nakládá jako s individuálními slovy. Pokud mu to vyjde tak, že se jedna část slova nachází na konci prvního řádku a druhá část na začátku druhého řádku, umístí za první řádek spojovník. Ano, chápete to správně, jedná se o jakési primitivní dělení slov. Má to ale jeden háček - musíte do

každého slova jako roboti vkládat měkké spojovníky mezi slabiky, což není zrovna ideální.

Tento měkký spojovník se za normálních okolností (kdy se celé slovo nachází na jednom řádku) nevykresluje, ani by neměl být brán v úvahu při indexování, vyhledávání apod. Nyní příklad:

```
<p>
Lorem ipsum <!-- Mezi „Lorem“ a „ipsum“ nesmí dojít k zalomení. --> dolor sit
amet, consectetuer adipiscing elit. Duis volutpat.<br /> <!-- Další text bude
pokračovat až na další řádce. --> Duis leo. Donec lao&shy;reet <!-- Toto slovo může
být rozděleno mezi „lao“ a „reet“, jinak nesmí být spojovník vykreslen. --> iaculis
quam. Vivamus ipsum. Vestibulum feugiat, neque pretium aliquet pulvinar.
</p>
```

## Element pre - přesné formátování textu

**Povolený obsah:** `%pre.content;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`xml:space (preserve) #FIXED 'preserve'`

Tento atribut udává, že interpret XHTML nesmí odstraňovat bílá místa uvnitř elementu. Hodnota tohoto atributu nemůže být jiná než implicitní `preserve`, proto ani nemá smysl ho používat.

Element pre v XHTML 1.0 Strict DTD

Text uvnitř tohoto blokového elementu musí být ponechán a interpretován **včetně bílých míst** - tedy různých zlomů řádků, mezer apod. Interpret XHTML také může použít písmo s pevnou šířkou znaků (což se dá změnit ve stylech) a nepoužívat zde automatické zalamování řádků, nevizuální interpret může také bílá místa ignorovat a význam tohoto elementu tak potlačit.

Nyní k tomu, co může element `pre` uzavírat. Jak vidíte, jeho **obsah** je deklarován na speciální parametrickou entitu `%pre.content;`. Její deklarace vypadá takto:

```
(#PCDATA | a | %fontstyle; | %phrase; | %special.pre; | %misc.inline; |
%inline.forms;)*
```

Nebudeme se s ní zatím příliš obšírně zabývat, pouze si řekneme, že se v podstatě jedná o nám již známou entitu `%Inline;`, pouze bez elementů `img` a `object` pro vkládání externích objektů do stránky. Také byste zde neměli používat dnes představené `sub`, `sup` a elementy `small` spolu s `big` (těmi se ještě budeme zabývat). Je to kvůli tomu, aby byla v tomto elementu zachována jakási pevná mřížka - tu byste neměli narušovat ani pomocí stylů.

# XHTML - seznamy a výčty

V podrobném přehledu XHTML budeme pokračovat seznamy a výčty, mimořádně důležitými prvky XHTML pro strukturování textu.

Pokud si nedovedete pod pojmem seznam

či výčet nic představit, potom věžte, že v XHTML jsou **seznamy neuspořádané**:

- položka
- další položka
- a ještě jedna položka

...a **uspořádané**:

1. první položka
2. druhá položka
3. třetí položka

Navíc se v XHTML setkáme ještě s třetím typem seznamu, a sice **seznamem definic**, nebo-li výčtem. Ten je od předchozích dvou trochu odlišný:

**pojem**
vysvětlení pojmu

**další pojem**
vysvětlení dalšího pojmu
jiné vysvětlení tohoto pojmu

Nyní se již ponoříme do podrobnějšího popisu.

## Elementy ul a ol – seznamy

**Povolený obsah:** `(li)+`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element ul v XHTML 1.0 Strict DTD

Element ol v XHTML 1.0 Strict DTD

Pomocí elementu `ul` se ohraničuje neuspořádaný seznam, pomocí `ol` uspořádaný seznam. Tyto elementy ale samy o sobě žádný seznam nevytvoří, to se dělá až pomocí elementů `li`, které uzavírají jednotlivé položky.

Neuspořádané seznamy byste měli využívat, pokud máte určitý výčet věcí, úkonů apod. a

chcete zdůraznit jejich rovnocennost. Uspořádaný seznam patří tam, kde je navíc třeba zdůraznit i pořadí (např. postup práce).

## Element li – položky seznamu

**Povolený obsah:** `%Flow;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element li v XHTML 1.0 Strict DTD

Elementy `li` ohraničují jednotlivé položky seznamu. Vedle jejich obsahu se automaticky vytváří tzv. odrážka, jejíž typ můžete změnit pomocí stylů. Do

tohoto elementu můžete uzavřít téměř libovolné blokové či řádkové elementy nebo text (díky parametrické entitě `%Flow;`, na níž je deklarován jeho obsah). Nyní se tedy již můžete podívat, pomocí jakého kódu jsme vytvořili první dva příklady v úvodu článku:

```
<ul>
  <li>položka</li>
  <li>další položka</li>
  <li>a ještě jedna položka</li>
</ul>


<ol>
  <li>první položka</li>
  <li>druhá položka</li>
  <li>třetí položka</li>
</ol>
```

## Seznamy definic

### Element dl – seznam definic

**Povolený obsah:** `(dt|dd)+`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element dl v XHTML 1.0 Strict DTD

Stejně jako `ul` a `ol`, i element `dl` pouze ohraničuje obsah seznamu. Samotné položky se zde ale narozdíl od předchozích seznamů vytvářejí pomocí elementů `dt` a `dd`:

### Element dt – definice

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`

Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element dt v XHTML 1.0 Strict DTD

### Element dd – popis

**Povolený obsah:** `%Flow;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element dd v XHTML 1.0 Strict DTD

Uvnitř seznamu se vždy vyskytuje definice reprezentovaná elementem `dt` následovaná jedním či více popisy, které jsou reprezentovány elementy `dd`.

Samozřejmě nemusíte seznamy definic používat jen k vysvětlování pojmů, využití může být obecnější – např. když chcete mluvit o několika podobných tématech, umístěte tato témata do elementu `dt` a vysvětlování do elementů `dd`.

Elementy `dt` smí obsahovat pouze text a řádkové elementy, proto by definice, která je zde uzavřená, měla být co nejkratší. Naopak do elementu `dd` můžeme díky `%Flow;` umístit text, řádkové i blokové elementy. Nyní si opět ukážeme, pomocí jakého kódu jsme vytvořili příklad použití seznamů definic v úvodu článku:

```
<dl>
  <dt>pojem</dt>
    <dd>vysvětlení pojmu</dd>
  <dt>další pojem</dt>
    <dd>vysvětlení dalšího pojmu</dd>
    <dd>jiné vysvětlení tohoto pojmu</dd>
</dl>
```

## Proč byste měli seznamy používat?

Používání seznamů je kromě správného strukturování stránky důležité i pro dobrou orientaci čtenáře na stránce. Jak již jistě víte, weboví uživatelé jsou většinou líní číst a dlouhé texty je snadno odradí. Proto je rozdělování delšího textu na kratší úseky pomocí nadpisů, seznamů apod. tak důležité.

## Kombinování seznamů

Na závěr ještě k tomu, jak lze seznamy navzájem kombinovat. Díky faktu, že položky seznamů mohou obsahovat téměř libovolné elementy, můžete seznamy nejrůznějším způsobem vnořovat do sebe. Např. můžete umístit uspořádaný seznam do neuspořádaného seznamu:

```
<ul>
  <li>text ...
    <ol>
      <li>text ... </li>
      <li>text ... </li>
```

```
    </ol>
  </li>
  <li>text ... </li>
</ul>
```

Stejně tak můžete i vnořit uspořádaný seznam do seznamu definic:

```
<dl>
  <dt>definice</dt>
    <dd>vysvětlení ...
      <ul>
        <li>text ... </li>
        <li>text ... </li>
      </ul>
    </dd>
  <dt>definice</dt>
    <dd>vysvětlení ...</dd>
</dl>
```

# XHTML - tabulky

Tímto článkem se začneme zabývat tabulkami. Z běžného života je jistě dobře znáte, používají se k prezentaci různých ceníků, srovnání a dalších "tabulkových dat". Jistě ale nevíte, jak takovou tabulku vložit do XHTML dokumentu...

Tabulky v XHTML dávají nám, autorům, možnost prezentovat na stránce data **dvojrozměrně** – uživatel může tato data vnímat jak v řádcích tak ve sloupcích. To je u mnoha typů dat poměrně vhodný a většinou nenahraditelný prostředek.

Z této "dvojrozměrnosti" však vyplývají i nevýhody, tabulka musí být prezentována ve dvojrozměrném výstupu (obrazovce) a pokud možno celá najednou. Bohužel ne všechna zařízení pro práci s webem (např. čtečky obrazovky) mají tyto schopnosti, a i když XHTML obsahuje prostředky ke zpřístupnění tabulky i v takových zařízeních, vždy je zde práce s tabulkou poměrně komplikovaná a zdlouhavá. Proto je lepší použití tabulky nejprve pořádně zvážit, a pokud je data možné reprezentovat jinou formou (např. pomocí výčtu), použít tuto formu nebo ji alespoň nabídnout jako alternativu k tabulce.

Kromě uspořádání tabulkových dat se tabulky někdy používají i k vytvoření vzhledu stránky – této praxe byste se ale již měli vyvarovat a přenechat vzhled stránky stylům.

## Z čeho se skládá tabulka

Každá tabulka v XHTML je tvořena svým *titulkem* a samotnými daty. Ta se nacházejí v takzvaných *buňkách*. Každá buňka patří minimálně do jednoho *řádku* a jednoho *sloupce*. Řádky se mohou seskupovat do *skupin řádků* a sloupce do *skupin sloupců*.

## Element table – ohraničení tabulky

**Povolený obsah:** `(caption?, (col*|colgroup*), thead?, tfoot?, (tbody+|tr+))`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`summary %Text; #IMPLIED`
V tomto atributu se má nacházet souhrn účelu a struktury tabulky pro nevizuální interprety XHTML, např. čtečku obrazovky.

`width %Length; #IMPLIED`
Nastavuje šířku tabulky.

`border %Pixels; #IMPLIED`
Tento atribut určuje šířku rámečku.

`frame %TFrame; #IMPLIED`

Tento atribut určuje, jaké strany rámečku kolem tabulky budou viditelné:

`void`
Žádná strana nebude viditelná – tak tomu je i pokud tento atribut nenastavíte.

`above`
Jenom horní strana orámování bude viditelná.

`below`

Jenom dolní strana orámování bude viditelná.

**`hsides`**
Jenom horní a dolní strana bude viditelná.

**`vsides`**
Viditelné budou jenom levá a pravá strana.

**`lhs`**
Viditelná bude jenom levá strana.

**`rhs`**
Viditelná bude jenom pravá strana.

**`box`** **nebo** **`border`**
Všechny strany budou viditelné.

```
rules       %TRules;       #IMPLIED
```

Tento atribut určuje, jaký způsob orámování se použije uvnitř tabulky (jak budou orámovány její buňky):

**`none`**
Uvnitř tabulky nebude žádné orámování.

**`groups`**
Orámováním od sebe budou odděleny skupiny řádek a skupiny sloupců (k těm se ještě dostaneme v dalších dílech).

**`rows`**
Orámováním od sebe budou odděleny řádky tabulky.

**`cols`**
Orámováním od sebe budou odděleny sloupce tabulky.

**`all`**
Orámováním od sebe budou odděleny všechny řádky i sloupce tabulky.

```
cellspacing       %Length;       #IMPLIED
```
Tento atribut nastavuje množství volného místa, které se má nacházet mezi buňkami tabulky.

```
cellpadding       %Length;       #IMPLIED
```
Tento atribut nastavuje množství volné místa mezi orámováním buňky a jejím obsahem.

Element table v XHTML 1.0 Strict DTD

Všechny prvky tabulky, jako jsou buňky nebo řádky, se nacházejí uvnitř elementu `table`. Ten může podle DTD obsahovat postupně:

1. nepovinně jeden element `caption`,
2. libovolné množství (i nula) elementů `col` nebo libovolné množství elementů `colgroup`,
3. nepovinně jeden element `thead`,
4. nepovinně jeden element `tfoot` a
5. jeden či více elementů `tbody` nebo jeden či více elementů `tr`.

Základní struktura tabulky by proto mohla vypadat například takto (nám zatím neznáme elementy jsou pouze naznačeny):

```
<table summary="Ukázková tabulka – obsahuje x sloupců a y řádků. První řádky
reprezentují...">
  <caption></caption>

  <col />
  <col />

  <thead></thead>
  <tfoot></tfoot>
```

```
  <tbody></tbody>
  <tbody></tbody>

</table>
```

**Poznámka:** Možná jste si všimli, že i ve striktní verzi XHTML lze využít u elementu `table` (i dalších tabulkových elementů, ke kterým se ještě dostaneme) mnoha atributů, které určují vzhled tabulky. Již ale víte, že k nastavení vzhledu stránky slouží styly, tyto atributy by tedy neměly do striktního XHTML patřit. Ve skutečnosti ale v době publikování HTML 4 ještě jazyk CSS neměl vlastní řešení vzhledu tabulek, proto byly tyto atributy ve striktním HTML ponechány. Dnes již však lze nastavit vzhled tabulek i v CSS, a proto byste vzhledové atributy tabulek neměli používat - já je zde budu uvádět pouze ve stručnosti a nebudeme se jimi příliš zatěžovat.

# XHTML - titulek, skupiny řádků a řádky tabulek

V následujícím článku si budeme povídat o titulku tabulky, ukážeme si jednotlivé skupiny řádek tabulky a také samotné řádky.

## Element caption - titulek tabulky

**Povolený obsah:** `%Inline;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

Element caption v XHTML 1.0 Strict DTD

Nepovinný element `caption` by měl obsahovat titulek (nadpis) tabulky. Tento titulek by měl sloužit hlavně k rychlému pochopení významu tabulky, delší popis struktury a účelu tabulky by měl obsahovat atribut `summary` elementu `table`.

Co se týče vizuálního výstupu, titulek tabulky se nachází obvykle na vrcholu tabulky, jeho umístění ale může být změněno ve stylech.

## Elementy thead, tfoot a tbody - skupiny řádků

**Povolený obsah:** `(tr)+`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`align (left|center|right|justify|char) #IMPLIED`
Tento atribut určuje zarovnání textu v buňkách tabulky, které patří do dané skupiny řádků. Hodnota `left` zarovná text doleva, `center` na střed, `right` doprava, `justify` do bloku („na obě strany“) a `char` zarovná buňky podle určitého znaku (např. desetinné čárky) - v takovém případě prohlížeč musí zohlednit i hodnoty atributů `char` a `charoff`.

`char %Character; #IMPLIED`
Určuje znak, podle kterého se má text zarovnat, v případě, že je `align="char"`.

`charoff %Length; #IMPLIED`
Nastavuje odsazení textu od levého okraje až ke znaku, podle kterého se text zarovnává (v případě, že je `align="char"`).

`valign (top|middle|bottom|baseline) #IMPLIED`
Určuje svislé zarovnání textu v buňkách tabulky, které jsou součástí dané skupiny řádků. Hodnota `top` zarovná text nahoru, hodnota `middle` na střed (výchozí hodnota), `bottom` dolů a `baseline` zaručí, že všechny buňky v jednom řádku budou mít první řádek textu zarovnán na stejnou základní linku.

Element thead v XHTML 1.0 Strict DTD

Element tfoot v XHTML 1.0 Strict DTD

Element tbody v XHTML 1.0 Strict DTD

Řádky v tabulce je možné rozdělit do skupin pomocí elementů `thead`, `tfoot` a `tbody`. Tyto elementy se liší svým významem, ale jsou u nich povoleny ty samé atributy a mají ten samý povolený obsah (jeden či více elementů `tr`, což je řádek tabulky).

Element `thead` označuje ty řádky tabulky, které tvoří její záhlaví. Jsou to obvykle buňky s názvy či popisky sloupců, neobsahují samotná data.

Element `tfoot` označuje ty řádky tabulky, které tvoří její zápatí. Zde se mohou nacházet například součty sloupců nebo jiný jejich souhrn.

Element `tbody` by měl obsahovat samotná data tabulky (*tělo*).

Elementy `thead` a `tfoot` se mohou v tabulce vyskytovat maximálně jednou, element `tbody` i vícekrát (rozdělit tabulku do více *těl* byste měli v případě, že se její části zabývají odlišnými tématy, mají jiný význam).

Zápatí tabulky (`tfoot`) navíc musí následovat hned za jejím záhlavím (`thead`), až za nimi se může nacházet tělo tabulky (`tbody`) - při zobrazení tabulky je ale pořadí záhlaví, tělo, zápatí. Tato zdánlivá nelogičnost je zapříčiněna pořadím nahrávání tabulky - interpret XHTML by měl nejdříve obdržet a zobrazit záhlaví a zápatí tabulky, a až poté mezi ním zobrazovat její tělo.

Informací o záhlaví a zápatí může interpret XHTML využít tak, že zobrazí tělo tabulky s posuvníkem, zatímco záhlaví a zápatí budou nad, respektive pod, tímto tělem a při jeho posouvání budou trvale viditelné. Stejně tak při tisku může být záhlaví a zápatí zopakováno na každé straně, na které je tabulka tisknuta.

## Element tr - řádek tabulky

**Povolený obsah:** `(th|td)+`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`align (left|center|right|justify|char) #IMPLIED`
Tento atribut zde má stejný význam jako u skupin řádků, pouze se používá pro buňky uvnitř daného řádku.

`char %Character; #IMPLIED`
Tento atribut zde má stejný význam jako u skupin řádků, pouze se používá pro buňky uvnitř daného řádku.

`charoff %Length; #IMPLIED`
Tento atribut zde má stejný význam jako u skupin řádků, pouze se používá pro buňky uvnitř daného řádku.

`valign (top|middle|bottom|baseline) #IMPLIED`
Tento atribut zde má stejný význam jako u skupin řádků, pouze se používá pro buňky uvnitř daného řádku.

Element tr v XHTML 1.0 Strict DTD

Element `tr` reprezentuje řádek tabulky, přičemž v sobě uzavírá buňky, které tento řádek tvoří (elementy `th` a `td`). Obvykle se nachází uvnitř jedné ze skupin řádek:

```
<table summary="...">
  <caption>...</caption>

  <thead>
    <tr> <!-- první řádek -->
```

```
      ...
    </tr>
  </thead>

  <tbody>
  <!-- tato tabulka neobsahuje zápatí -->
    <tr> <!-- druhý řádek -->
      ...
    </tr>
    <tr> <!-- třetí řádek -->
      ...
    </tr>
  </tbody>

</table>
```

Element `tr` však může stát i osamoceně (toto řešení je vhodné hlavně u krátkých tabulek, které nepotřebují rozdělení na záhlaví, zápatí a tělo):

```
<table summary="...">
  <caption>...</caption>

  <tr> <!-- první řádek -->
    ...
  </tr>

  <tr> <!-- druhý řádek -->
    ...
  </tr>

  <tr> <!-- třetí řádek -->
    ...
  </tr>

</table>
```

# XHTML - buňky tabulky

Dnes si budeme povídat o nejdůležitější součásti tabulek, a sice jejích buňkách. Ukážeme si, jak je používat i jak pomocí některých jejich atributů zpřístupnit tabulku pro nevizuální prohlížeče.

## Elementy th a td - buňky tabulky

**Povolený obsah:** `%Flow;`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

**`abbr %Text; #IMPLIED`**
V tomto atributu by se měla nacházet zkrácená podoba obsahu buňky, pokud je její obsah delší. To se týká hlavně hlavičkových buněk, jejichž obsah může být při nevizuálním výstupu mnohokrát opakován. Hodnotu tohoto atributu může ale použít i vizuální prohlížeč a vykreslit ji jako obsah elementu, v případě že v buňce nemá dostatek místa na vykreslení jejího celého obsahu.

**`axis CDATA #IMPLIED`**
Pomocí tohoto atributu může být buňka přiřazena do určité kategorie, jeho hodnotou jsou čárkou oddělená jména kategorií. K tomuto tématu se dnes ještě dostaneme.

**`headers IDREFS #IMPLIED`**
Tento atribut obsahuje mezerou oddělená ID-jména těch hlavičkových buněk, které patří k dané buňce. Vše si ještě detailněji vysvětlíme.

**`scope (row|col|rowgroup|colgroup) #IMPLIED`**

Tento atribut je jednodušší alternativou atributu `headers`. Nastavuje se u hlavičkových buněk a určuje, ke kterým buňkám s daty daná hlavičková buňka patří. Může nabývat jedné z těchto hodnot:

**`row`**
Daná buňka je hlavičková pro buňky, které následují v řádku za ní.

**`col`**
Daná buňka je hlavičková pro buňky, které následují v sloupci za ní.

**`rowgroup`**
Daná buňka je hlavičková pro buňky, které následují ve skupině řádků za ní.

**`colgroup`**
Daná buňka je hlavičková pro buňky, které následují ve skupině sloupců za ní.

**`rowspan %Number; "1"`**
Tento atribut říká, na kolika řádcích se daná buňka rozpíná. Výchozí hodnotou je 1, hodnota 0 znamená, že buňka se rozpíná přes všechny řádky od toho, v kterém se nachází, až po konec skupiny řádků (`thead`, `tbody` a `tfoot`), v které se nachází.

**`colspan %Number; "1"`**
Tento atribut říká, v kolika sloupcích se daná buňka rozpíná. Výchozí hodnotou je opět 1, hodnota 0 znamená, že buňka se rozpíná přes všechny sloupce od toho, v kterém se nachází, až po konec skupiny sloupců (element `colgroup`, kterým se ještě budeme zabývat), v které se nachází.

**`align (left|center|right|justify|char) #IMPLIED`**
Tento atribut určuje vodorovné zarovnání obsahu v buňce, má zde stejný význam jako u skupin řádků.

**`char %Character; #IMPLIED`**

Tento atribut nastavuje znak, podle kterého se má obsah buňky zarovnat. Také jsme se jím již zabývali u skupin řádků.

```
charoff       %Length;       #IMPLIED
```

Tento atribut určuje odsazení zarovnávacího znaku, má také stejný význam jako u skupin řádků.

```
valign       (top|middle|bottom|baseline)       #IMPLIED
```

Atribut `valign` určuje svislé zarovnání obsahu v buňce, také jsme se jím již zabývali.

Element th v XHTML 1.0 Strict DTD

Element td v XHTML 1.0 Strict DTD

V XHTML můžeme použít dva typy buněk v tabulce:

- **hlavičkové**, ve kterých se nachází nějaké hlavičkové informace (např. názvy produktů), jsou reprezentovány elementem `th` a ve vizuálních prohlížečích je jejich obsah obvykle vykreslen tučným písmem (to lze ale změnit ve stylech) a
- **datové**, ve kterých se nachází samotná data (např. množství prodaných produktů) a jsou reprezentovány elementem `td`.

U obou typů hlaviček můžeme použít ty samé atributy, i když ne všechny vždy dávají smysl. Atributy `abbr`, `axis` a `scope` naleznou své využití hlavně u hlavičkových buněk, zatímco `headers` u datových buněk.

Buňky mohou být i prázdné (pomocí zápisu `<td></td>` nebo `<th></th>`). Styly v takovém případě určují, jestli budou tyto buňky zobrazeny.

Buňky mohou obsahovat téměř libovolné další elementy i text, jejich obsah je velmi volný (vzhledem k parametrické entitě `%Flow;`).

### Příklad

Nyní již konečně máme dostatek znalostí na to, abychom si mohli ukázat první kompletní tabulku. Bude se jednat o přehled cen zájezdů do různých zemí v různých měsících. Pod těmito cenami se bude nacházet vždy ta nejlevnější nabídka:

```
<table summary="Tato tabulka obsahuje ceny našich zájezdů. V prvním řádku se
nacházejí země, kam se můžete vydat, v prvním sloupci se nacházejí měsíce, ve kterých
lze zájezd uskutečnit. Poslední řádek obsahuje nejlevnější cenu zájezdu ze všech
měsíců.">
  <caption>Ceny zájezdů</caption>

  <thead>
    <tr>
      <th></th>
      <th>Anglie</th>
      <th>USA</th>
      <th>Maledivy</th>
    </tr>
  </thead>

  <tfoot>
    <tr>
      <th>Nejlevnější</th>
      <td>7 000 Kč</td>
      <td>30 000 Kč</td>
      <td>40 000 Kč</td>
    </tr>
  </tfoot>

  <tbody>
    <tr>
      <th>Květen</th>
```

```
      <td>10 000 Kč</td>
      <td>30 000 Kč</td>
      <td>40 000 Kč</td>
    </tr>
    <tr>
      <th>Červenec</th>
      <td>15 000 Kč</td>
      <td>40 000 Kč</td>
      <td>65 000 Kč</td>
    </tr>
    <tr>
      <th>Září</th>
      <td>7 000 Kč</td>
      <td>35 000 Kč</td>
      <td>50 000 Kč</td>
    </tr>
  </tbody>

</table>
```

Všimněte si:

- jakým způsobem jsme použili atribut `summary` elementu `table` a element `caption`,
- jak jsme rozčlenili tabulku na části pomocí elementů `thead`, `tfoot` a `tbody`,
- že hlavičkové buňky vyznačujeme pomocí `th`, datové pomocí `td` a
- že uvnitř čísel a mezi číslem a zkratkou měny se nachází nedělitelná mezera, o které jsme mluvili již na začátku našeho seriálu - to zabrání prohlížeči rozdělit tyto hodnoty na více řádků při vizuálním výstupu.

Nyní se podívejte na to, jak takovou tabulku zobrazí vizuální prohlížeč po přiřazení několika stylů:

Nabídka zájezdů

| | Anglie | USA | Maledivy |
|---|---|---|---|
| Květen | 10 000 Kč | 30 000 Kč | 40 000 Kč |
| Červenec | 15 000 Kč | 40 000 Kč | 65 000 Kč |
| Září | 7 000 Kč | 35 000 Kč | 50 000 Kč |
| Nejlevnější | 7 000 Kč | 30 000 Kč | 40 000 Kč |

## Atributy rowspan a colspan

Nyní si ukážeme efekt atributů `rowspan` a `colspan` na buňky tabulky. Pozměníme trochu záhlaví tabulky z předchozího příkladu:

```
  <thead>
    <tr>
      <th></th>
      <th colspan="2">Anglie</th>
      <th rowspan="2">USA</th>
      <th rowspan="2">Maledivy</th>
    </tr>
    <tr>
      <th></th>
      <th title="sever Anglie">sever</th>
      <th title="jih Anglie">jih</th>
    </tr>
  </thead>
```

Tímto krokem jsme buňku „Anglie“ rozšířili do dvou sloupců, protože se v řádku pod ní má nacházet ještě rozdělení zájezdů na sever Anglie a jih Anglie. Buňky „USA“ a „Maledivy“ ale žádné rozdělené nepotřebují, proto jsme je roztáhli do dvou řádků. Samozřejmě jsme do těla a zápatí tabulky museli přidat ještě jeden sloupec, protože do Anglie nyní máme dva druhy zájezdů. Obrázek pomůže v pochopení:

**Nabídka zájezdů**

| | Anglie | | USA | Maledivy |
|---|---|---|---|---|
| | sever | jih | | |
| Květen | 10 000 Kč | 11 000 Kč | 30 000 Kč | 40 000 Kč |
| Červenec | 15 000 Kč | 17 000 Kč | 40 000 Kč | 65 000 Kč |
| Září | 7 000 Kč | 7 500 Kč | 35 000 Kč | 50 000 Kč |
| Nejlevnější | 7 000 Kč | 7 500 Kč | 30 000 Kč | 40 000 Kč |

Vždy byste měli dbát na to, aby v žádném řádku tabulky nechyběla ani nepřebývala žádná buňka - tedy aby tabulka „přesně pasovala“. Zajistit to můžete pomocí vhodné kombinace elementů `td`, `th` (i prázdných) a jejich atributů `colspan` a `rowspan`.

## Atributy pro zpřístupnění tabulky v nevizuálním prohlížeči

Nejprve si předvedeme, jak by vypadal kód prvního příkladu v případě, že bychom ho chtěli zpřístupnit pomocí atributu `scope`:

```
<table summary="Tato tabulka obsahuje ceny našich zájezdů...">
  <caption>Ceny zájezdů</caption>

  <thead>
    <tr>
      <th></th>
      <th scope="col">Anglie</th>
      <th scope="col">USA</th>
      <th scope="col">Maledivy</th>
    </tr>
  </thead>

  <tfoot>
    <tr>
      <th scope="row">Nejlevnější</th>
      <td>7 000 Kč</td>
      <td>30 000 Kč</td>
      <td>40 000 Kč</td>
    </tr>
  </tfoot>

  <tbody>
    <tr>
      <th scope="row">Květen</th>
      <td>10 000 Kč</td>
      <td>30 000 Kč</td>
      <td>40 000 Kč</td>
    </tr>
    <tr>
      <th scope="row">Červenec</th>
      <td>15 000 Kč</td>
      <td>40 000 Kč</td>
      <td>65 000 Kč</td>
    </tr>
```

```
    <tr>
      <th scope="row">Září</th>
      <td>7 000 Kč</td>
      <td>35 000 Kč</td>
      <td>50 000 Kč</td>
    </tr>
  </tbody>

</table>
```

Hlavičkovým buňkám jsme přiřadili takovou hodnotu atributu `scope`, aby nevizuální interpret XHTML mohl určit pro každou datovou buňku i hlavičkové buňky, které k ní patří. U každé datové buňky potom může uživatele obeznámit i s hlavičkovými buňkami, které k ní náleží.

Atribut `scope` je vhodný pro zpřístupnění jednoduchých tabulek, jako je ta z prvního příkladu. V praxi bychom se rozhodli v tomto případě určitě pro tento atribut. Z demonstračních účelů vám ale nyní předvedu zpřístupnění tabulky i pomocí atributu `headers`:

```
<table summary="Tato tabulka obsahuje ceny našich zájezdů...">
  <caption>Ceny zájezdů</caption>

  <thead>
    <tr>
      <th></th>
      <th id="hlavicka1">Anglie</th>
      <th id="hlavicka2">USA</th>
      <th id="hlavicka3">Maledivy</th>
    </tr>
  </thead>

  <tfoot>
    <tr>
      <th id="hlavicka4">Nejlevnější</th>
      <td headers="hlavicka1 hlavicka4">7 000 Kč</td>
      <td headers="hlavicka2 hlavicka4">30 000 Kč</td>
      <td headers="hlavicka3 hlavicka4">40 000 Kč</td>
    </tr>
  </tfoot>

  <tbody>
    <tr>
      <th id="hlavicka5">Květen</th>
      <td headers="hlavicka1 hlavicka5">10 000 Kč</td>
      <td headers="hlavicka2 hlavicka5">30 000 Kč</td>
      <td headers="hlavicka3 hlavicka5">40 000 Kč</td>
    </tr>
    <tr>
      <th id="hlavicka6">Červenec</th>
      <td headers="hlavicka1 hlavicka6">15 000 Kč</td>
      <td headers="hlavicka2 hlavicka6">40 000 Kč</td>
      <td headers="hlavicka3 hlavicka6">65 000 Kč</td>
    </tr>
    <tr>
      <th id="hlavicka7">Září</th>
      <td headers="hlavicka1 hlavicka7">7 000 Kč</td>
      <td headers="hlavicka2 hlavicka7">35 000 Kč</td>
      <td headers="hlavicka3 hlavicka7">50 000 Kč</td>
    </tr>
  </tbody>

</table>
```

Jak vidíte, tento způsob je o dost náročnější, hlavně co se psaní týče. U složitějších tabulek již ale nemáte většinou na výběr, protože jejich struktura neumožňuje použití atributu `scope`.

Co jsme tedy udělali? Nejprve jsme si pojmenovali pomocí atributu `id` všechny hlavičkové buňky a poté jsme u každé datové buňky pomocí atributu `headers` určili, k jakým hlavičkovým buňkám tato datová buňka patří (použili jsme zde hodnoty atributu `id` těchto buněk).

### Atribut axis

Pomocí atributu `axis` můžeme přiřadit každou hlavičkovou buňku do nějaké kategorie (kterou si sami vytvoříme). Např. buňky „Anglie“, „USA“ a „Maledivy“ z našeho prvního příkladu bychom mohli přiřadit do kategorie „Země“ a buňky „Květen“, „Červenec“ a „Září“ do kategorie „Měsíce“.

Díky tomu budeme u každé datové buňky znát její hlavičkové buňky i kategorie, do kterých tyto buňky patří. To může počítači pomoct v prezentaci tabulky i v jejím zpracování.

Více o atributu `axis` i o ostatních „zpřístupňovacích“ atributech (`headers`, `scope` a `abbr`) se dozvíte v mém článku Odstraňte bariéry svého webu - tabulky. Tento článek vám doporučuji i v případě, že máte potíže s pochopením významu těchto atributů.

# XHTML - sloupce

V tomto článku série o XHTML si řekneme něco o sloupcích v tabulce - představíme si elementy colgroup a col.

Asi jste si již stačili všimnout, že v XHTML tabulce stačí nadefinovat pouze řádky tabulky - sloupce pak vzniknou automaticky. Někdy se ale hodí označit nějak i sloupce tabulky - jednak i v nich může být jistá struktura (různé sloupce se mohou týkat různého tématu), jednak se často hodí zvýraznit vybrané z nich pomocí stylů.

Právě v takové chvíli přichází na pomoc elementy `colgroup` a `col`. Jak již víte, zapisují se v tabulce hned za element `caption`. Narozdíl od řádků se do nich ale nevnořují buňky tabulky (ty se v XHTML nacházejí pouze uvnitř řádků). Tyto elementy jsou jednoduše prázdné a jejich jediným účelem je aplikovat určité společné vlastnosti na sloupec (či sloupce).

V následujícím textu se naučíte elementy `colgroup` a `col` používat.

## Element colgroup - skupiny sloupců

**Povolený obsah:** `(col)*`

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`span %Number; "1"`
Tento atribut nastavuje, kolik sloupců daná skupina obsahuje. Jeho hodnotou musí být celé číslo větší než nula, výchozí hodnotu `1` není třeba nastavovat. Pokud daný element `colgroup` obsahuje nějaké elementy `col` (viz dále), určují počet sloupců tyto elementy a obsah atributu `span` je ignorován.

`width %MultiLength; #IMPLIED`
Tento atribut určuje šířku každého ze sloupců, které skupina obsahuje. Pro tento účel byste již ale dnes měli používat styly.

`align (left|center|right|justify|char) #IMPLIED`
Tento atribut určuje vodorovné zarovnání obsahu v buňkách, které se nachází v dané skupině sloupců. Má zde stejný význam jako u skupin řádků.

`char %Character; #IMPLIED`
Tento atribut nastavuje znak, podle kterého se má zarovnat obsah v buňkách, které patří do daného sloupce. Také jsme se jím již zabývali u skupin řádků.

`charoff %Length; #IMPLIED`
Atribut `charoff` určuje odsazení zarovnávacího znaku, má také stejný význam jako u skupin řádků.

`valign (top|middle|bottom|baseline) #IMPLIED`
Atribut `valign` určuje svislé zarovnání obsahu v buňkách, které patří do dané skupiny sloupců. Také jsme se jím již zabývali.

Element colgroup v XHTML 1.0 Strict DTD

Element `colgroup` vytváří skupinu sloupců v tabulce. Počet sloupců, které do této skupiny patří, se určuje buď pomocí atributu `span` nebo pomocí elementů `col`, které se uvnitř `colgroup` nacházejí.

Sloupce se počítají samozřejmě od prvního. Pokud tedy chcete pojmenovat skupinu od desátého k dvacátému sloupci, musíte ještě před ni vložit skupinu o devíti sloupcích:

```
<table ...>
  <caption>...</caption>
```

```
  <colgroup span="9"></colgroup>
  <colgroup span="11" id="nase-sloupce"></colgroup>

  ...

</table>
```

**Poznámka:** Protože v tomto příkladu neobsahují elementy `colgroup` žádný obsah, bylo by je možné ukončit i pomocí zápisu `<colgroup />`. Elementy `colgroup` ale nějaký obsah mít mohou, a proto by s tímto zápisem mohly mít některé prohlížeče problémy - z toho důvodu je lepší zápis, který jsem použil v příkladu.

Element `colgroup` je element se **strukturálním významem** - to znamená, že byste pomocí něj neměli sdružovat sloupce tak, jak se vám to zrovna hodí. Sdružené sloupce by měly mít jakousi logickou spojitost, měly by se týkat stejného tématu.

Např. pokud budu mít ceník s položkami „hrušky“, „jablka“, „mrkve“ a „cibule“, potom by měla první skupina obsahovat první dvě položky (protože se jedná o ovoce) a druhá skupina třetí a čtvrtou položku (protože se jedná o zeleninu). Pokud potřebujete označit sloupce nezávisle na jejich struktuře, měli byste k tomu použít element `col`.

## Element col - sloupce

**Povolený obsah:** `EMPTY` (*žádný obsah*)

**Atributy:**

`%attrs;`
Tato parametrická entita obsahuje další parametrické entity: `%coreattrs;` (atributy `id`, `class`, `style` a `title`), `%i18n;` (atributy `lang`, `xml:lang` a `dir`) a `%events;` (atributy `onclick`, `ondblclick`, `onmousedown`, `onmouseup`, `onmouseover`, `onmousemove`, `onmouseout`, `onkeypress`, `onkeydown` a `onkeyup`).

`span %Number; "1"`
Tento atribut nastavuje, kolik sloupců daný element `col` zahrnuje. Jeho hodnotou musí být celé číslo větší než nula, výchozí hodnotu `1` není třeba nastavovat.

`width %MultiLength; #IMPLIED`
Tento atribut určuje šířku sloupců, které daný element `col` zahrnuje. Pro tento účel byste již ale dnes měli používat styly.

`align (left|center|right|justify|char) #IMPLIED`
Tento atribut určuje vodorovné zarovnání obsahu v buňkách, které patří do sloupců zahrnutých pomocí daného elementu `col`. Má zde stejný význam jako u skupin řádků.

`char %Character; #IMPLIED`
Tento atribut nastavuje znak, podle kterého se má zarovnat obsah v buňkách, které patří do sloupců zahrnutých pomocí daného elementu `col`. Také jsme se jím již zabývali u skupin řádků.

`charoff %Length; #IMPLIED`
Atribut `charoff` určuje odsazení zarovnávacího znaku, má také stejný význam jako u skupin řádků.

`valign (top|middle|bottom|baseline) #IMPLIED`
Atribut `valign` určuje svislé zarovnání obsahu v buňkách, které patří do sloupců zahrnutých pomocí daného elementu `col`. Také jsme se jím již zabývali.

Element col v XHTML 1.0 Strict DTD

Element `col` je velmi podobný elementu `colgroup`. Narozdíl od něj však nenese **žádnou strukturální informaci** - dá se říct, že jeho význam je podobný jako význam elementů `div` a `span`. Slouží tedy hlavně k označování sloupců pro účely stylů a skriptů, ne k logickému seskupování sloupců.

Element `col` se může nacházet přímo uvnitř elementu `table` (v takovém případě již nesmí být u této tabulky použity elementy `colgroup`) nebo uvnitř skupiny sloupců (`colgroup`).

V následujícím příkladu rozdělíme pět sloupců tabulky na první a druhé (díky tomu budeme moct pomocí stylů vytvořit „pruhované“ zobrazení):

```
<table ...>
  <caption>...</caption>

  <col class="prvni" />
  <col class="druhy" />
  <col class="prvni" />
  <col class="druhy" />
  <col class="prvni" />

  ...

</table>
```

V tomto příkladu uděláme to samé, sloupce v tabulce ale navíc rozdělíme pomocí skupin sloupců:

```
<table ...>
  <caption>...</caption>

  <colgroup id="tabulka-sloupce-ovoce">
    <col class="prvni" />
    <col class="druhy" />
  </colgroup>
  <colgroup id="tabulka-sloupce-zelenina">
    <col class="prvni" />
    <col class="druhy" />
  </colgroup>

  ...

</table>
```

## Shrnutí

Pokud máte tabulku, kde se nachází více tématických skupin sloupců, zapište tuto strukturu pomocí elementů `colgroup`. Pokud potřebujete specifické sloupce označit pro potřeby stylů, skriptů apod. a toto označené nesouvisí se strukturou tabulky, udělejte to pomocí elementu `col`.