KAPITOLA 3

Základy CSS a HTML

Jazyk HTML a kaskádové styly (CSS) musí bezpečně ovládat každý vývojář webu. Jazyk HTML se dá naučit poměrně snadno, ovšem pokud chcete zvládnout tipy a triky pro práci s CSS, potřebujete praxi. Teorie je v tomto případě méně, problém činí dostatek praxe. Bohužel každý internetový prohlížeč i dnes ještě interpretuje příkazy CSS jinak, což dokáže řadu vývojářů webu doslova rozpálit doběla.

Rozsah této knihy mi neumožňuje zabývat se příliš podrobně správným používáním kaskádových stylů. I přesto, že jsem si vědoma této nedokonalosti či neúplnosti, chtěla bych se pokusit napsat alespoň drobný úvod ke kaskádovým stylům. Při provozování diskuzního fóra se ukázalo, že weby pomocí systému Joomla! vytvářejí i uživatelé, kteří v oblasti HTML a CSS nemají žádné znalosti. Ocitl se zde například předseda fotbalového klubu nebo zapálený člen místní církevní obce.

V této kapitole se budu snažit v každém okamžiku zaměřovat na klasické příklady systému Joomla!. Tuto kapitolu si tedy určitě přečtěte i v případě, že se považujete v oboru jazyka HTML a kaskádových stylů za odborníky.

Z pozadí vývoje

Bez nadsázky lze říci, že celý Internet hovoří jazykem HTML. Jazyk HTML se řadí mezi stránkový popisný jazyk a je starý stejně jako Internet samotný. Tento jazyk vyvinul v roce 1990 Tim Berners Lee. Tehdy mu však vůbec nešlo o uspořádání informací, ale pouze o čistou strukturu obsahu. Proto se dají pomocí jazyka HTML zobrazovat pouze významová rozvržení obsahu. Dokumentům je možné nastavovat hierarchii nadpisů, je možné vytvářet seznamy, v jazyku HTML ale je možné zpracovávat i seznamy v tabulkách. Tehdy žádné možnosti rozložení obsahu neexistovaly.

Je zajímavé, že informace, které se daly před lety na Internetu najít, byly pro mnoho osob daleko přístupnější, než je tomu u mnohých dokumentů dnes.

K omezení přístupu pro osoby se znevýhodněním došlo v mnoha oblastech až s požadavkem na úpravu vzhledu publikovaných informací.

Tabulky, které byly původně určeny pro prezentování datových struktur, byly doslova zneužity k umisťování internetového obsahu na internetové stránky. Sémantické zvýrazňování nadpisů bylo značně zanedbáváno, protože se řádky tabulek daly opticky přizpůsobovat daleko snáze.

Kaskádové styly (CSS) jsou druhem jazyka pro formátování určeného pro HTML. Když jsem se v roce 1999 začala zabývat návrhem webů, byla jednou z prvních knih věnovaných CSS kniha *Cascading Style Sheets* od Erica Meyera. Dost jsem s ní tehdy pracovala a už po několika týdnech studia jsem zkonstatovala, že ten správný čas pro nasazení CSS tehdy ještě nenastal, a to zejména kvůli chybějící podpoře ze strany internetových prohlížečů. Proto mi tenkrát nezbylo nic jiného než se naučit provádět rozvržení obsahu pomocí tabulek.

Dnes je naštěstí situace úplně jiná. Díky používání CSS nestojí oddělování obsahu od jeho rozvržení v cestě vůbec nic.

Chcete-li dnes vytvářet výkonné internetové stránky vyhovující standardům, je používání CSS naprosto nezbytné a v této oblasti potřebujete dostatek znalostí i pro vytváření rozložení na stránkách ze systému Joomla!.

Problematikou HTML a CSS se zabývá řada vynikajících knih. V této knize mohu poskytnout pouze obecný přehled, kde se dozvíte, co se vůbec pod zkratkami HTML a CSS skrývá a jak se tyto technologie efektivně používají.

- *Tvorba WWW stránek pro úplné začátečníky*, autor: Martin Domes. Computer Press, 2008.
- *HTML a CSS – Krok za krokem,* autor: Faithe Wempen. Computer Press, 2007.
- *HTML, XHTML a CSS – Názorný průvodce tvorbou WWW stránek*, autor: Elizabeth Castro. Computer Press, 2007.
- *333 tipů a triků pro CSS – 2. aktualizované vydání*, autor: Martin Domes. Computer Press, 2011.
- *Mistrovství v CSS,* autoři: Dan Rubin, Ian Lloyd, Jeff Croft. Computer Press, 2007.

Jakou verzi HTML používat?

V praxi se dosti často řeší poměrně složitá otázka, na kterou je obtížné odpovědět. Jde o otázku, jakou verzi značkovacího jazyka (X)HTML používat. K dispozici je pouze HTML 4.01 nebo XHTML 1.0, přičemž od každé jsou tři verze – Transitional, Strict a Frameset. Horkou novinkou pak je HTML5.

Rozdíl mezi HTML 4.01 a XHTML 1.0 se často přeceňuje – co se týče rozsáhlosti jazyka výkonnosti, jsou prakticky totožné a liší se pouze drobnostmi. Mezi jazyky HTML 4.01 Strict a XHTML 1.0 je možné kdykoliv provést konverzi.

Systém Joomla! vytváří výstup v jazyce HTML založený na XHTML, takže v tomto případě je naprosto zbytečné ptát se, jaký jazyk použít.

Daleko důležitější je pro oba jazyky rozdíl mezi verzí *Strict* a *Transitional*. Verze Transitional totiž umožňuje používání již nedoporučovaných nebo neschválených parametrů.

Předpokládám, že většina čtenářů se již někdy setkala s pojmem *deprecated*. Výraz *deprecated* znamená „zavržený". Mezi tyto nežádoucí parametry kupříkladu patří parametry `vspace` či `hspace` pro prvek Image, nebo parametr `align`.

V souvislosti se systémem Joomla! bych vám chtěla doporučit variantu Transitional, a to i přesto, že standardní výstup ze systému Joomla! umožňuje použít variantu Strict.

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

Možná vám to přijde poněkud zvláštní, protože varianta Strict by byla daleko elegantnější volbou a také by více odpovídala standardům. Verzi Transitional vám doporučuji kvůli textovým editorům, které jsou součástí systému Joomla! a jsou velmi často používány.

Tyto textové editory totiž bohužel stále používají zavržené parametry. Validátor kódu HTML, je nástroj prověřující správnost vámi vytvořené internetové stránky. Ten by vám tyto „chybně použité" parametry dozajista vytkl.

Pokud se však rozhodnete výše uvedené textové editory nepoužívat, chcete-li upravovat zdrojový kód ručně, popřípadě chcete použít editor Xstandard, je samozřejmě lepší použít verzi Strict:

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

HTML5

Celý web v současnosti míří mílovými kroky k HTML5. Již nyní se můžete na Internetu setkat se spoustou internetových stránek v jazyce HTML5.

Zkuste si sami nějaké najít. Určitě budete úspěšní na internetové adrese *http://html5gallery.com/*. Verzi HTML5 nyní využívají weby zejména velkých společností, jako je kupříkladu Plus nebo Google.

Mě osobně nejvíce nadchla deklarace tohoto typu dokumentu:

```
<!Doctype HTML>
```

Konečně zas jednou něco jednoduchého!

Velmi zajímavé jsou sémanticky významné elementy. Hned mě napadají například elementy `nav` pro navigaci, `header`, `footer`, `aside` pro doplňující informace, `section` pro oblasti a `article` pro texty. Už jen tyto samotné elementy představují velké obohacení jazyka.

Se všemi těmito elementy se internetové prohlížeče Opera, Firefox, Safari (v nejnovějších verzích) a Internet Explorer 9 vypořádávají naprosto z problémů, zatímco Internet Explorer 6, 7 a 8 nikoliv. Elementy se nedají volat kaskádovými styly ani stylováním – tyto internetové prohlížeče je prostě nerozpoznají. Tento problém má dvě řešení. Buď okolo těchto elementů vložíte klasický element `div` a provedete příslušné stylování, nebo vložíte neznámé elementy do DOM prohlížeče pomocí JavaScriptu.

```
<!--[if lt IE 9]>
<script type="text/javascript">
document.createElement('section');
document.createElement('nav');
document.createElement('article');
document.createElement('header');
document.createElement('footer');
document.createElement('aside');
document.createElement('figure');
document.createElement('legend');
</script> <![endif]-->
```

Nevýhodou je samozřejmě skutečnost, že tento způsob je funkční pouze tehdy, pokud je v internetovém prohlížeči povolen JavaScript.

V šabloně beez5 si můžete ve správcovské části zvolit pro zobrazování kódu buď značkovací jazyk XHTML, nebo HTML5. Tato volba se realizuje pomocí tzv. Template Override, jejíž princip funkce blíže objasním později.

Základní kostra HTML

Základní kostru HTML své internetové stránky si ve své šabloně určujete sami. Tato kostra určuje obecnou strukturu vaší internetové stránky a říká, kde se uvnitř dokumentu budou nacházet jednotlivé části stránky, jako je hlavička, patička stránky apod. Tato základní kostra se dá následně skvěle dolaďovat pomocí CSS. Samotný systém Joomla! spravuje pouze dynamický obsah s předem strukturovaným zdrojovým kódem.

Tento zdrojový kód se ve verzi 1.6 výrazně změnil, zmodernizoval a uvedl na současný technologický standard. Více se budou tomuto tématu a stavbě zmiňované kostry věnovat později.

Stručný úvod do CSS

Úvod

Kaskádové styly probrané v jedné kapitole? To je naprosto nemožné. Kaskádovým stylům je věnována spousta jednotlivých knih, v některých případech jsou to dokonce celé série knih. Je tedy jasné, že se mi v jedné kapitole, či dokonce jen v jedné její části nemůže podařit vysvětlit princip fungování kaskádových stylů. Mým cílem je poskytnout vám základní poznatky týkající se kaskádových stylů a na následujících řádcích se o to chci pokusit.

Kaskádové styly jsou vlastně popisným jazykem strukturovaných dokumentů. Když použijete kaskádové styly, můžete přímo ovlivňovat uspořádání vytvořeného kódu HTML. Pomocí kaskádových stylů můžete formátovat nadpisy, seznamy či odstavce, popřípadě uspořádat obsah internetové stránky na monitor – to však je už něco jako vysoká škola využívání kaskádových stylů.

Kompletní informace týkající se kaskádových stylů naleznete na internetové adrese *http://www.w3.org/TR/2011/REC-CSS2-20110607/* .

Integrace příkazů CSS

Příkazy CSS můžete do své internetové stránky integrovat různými způsoby. Jejich důležitost ve všech deklaracích CSS závisí na jejich umístění v rámci dokumentu. Právě tato funkce dala kaskádovým stylům název.

Externí připojení v podobě souboru nebo jejich přímé uvedení v hlavičce dokumentu

První možnost spočívá v uložení údajů o stylování do externího souboru:

```
<link rel="stylesheet" href="/templates/beez/css/layout.css" type="text/css"
media="screen,projection" title="" />
```

Výše uvedený příkaz se uvádí do hlavičky internetové stránky. Jedná se o nejčastěji používanou možnost a také o možnost nejefektivnější, neboť umožňuje od sebe naprosto oddělit obsah a vzhled. Pokud veškeré údaje týkající se stylů uložíte do nějakého externího souboru, můžete při případných dalších úpravách internetových stránek změnit vzhled, aniž byste museli jakkoli měnit samotný zdrojový kód.

Do výše uvedené cesty můžete uvést libovolný počet souborů CSS. Použít se to dá tehdy, pokud chcete od sebe oddělit umístění obsahu od barevného podání. Integrovat je možné i speciální soubory stylů určené pro Internet Explorer 6, popřípadě vytvářet soubory určené přímo pro tisk. Když se tedy podíváte na soubor `index.php` ze šablony Beez, uvidíte v něm spoustu souborů CSS.

Další možností je uvedení údajů o stylech přímo do hlavičky internetové stránky. Tento způsob dosud používá řada volně dostupných rozšíření určených pro systém Joomla!, a to pro integraci speciálních šablon stylů.

```
<style type="text/css">
<!-- div.mycomponentdiv
{ background:#000}
}
-->
</style>
```

Výše uvedené příkazy stylů z externích rozšíření pro systém Joomla! se často k vlastnímu rozvržení nehodí a možná by bylo daleko lepší je vůbec nepoužívat. Vzhledem k tomu, že se objevují ve zdrojovém kódu po našem vlastním CSS, které jsme integrovali pomocí značky `link rel`, mají vyšší prioritu, takže se zde opět projeví kaskádování.

Pokud bychom však tyto styly umístili před naším kaskádovým stylem vloženým pomocí značky `link rel`, pak by žádný problém nevznikl.

Ještě horší je případ, když externí komponenty používají řádkové styly. Pak jsme, co se týče jejich změny, naprosto bez šance, neboť řádkové styly mají vyšší prioritu a přepisují všechny ostatní.

Řádkové styly

Elementy HTML mají k dispozici parametr `style`, pomocí něhož je možné provádět jejich formátování:

```
<p style="color:#cc0000"> Toto je odstavec psaný červeným písmem.</p>
```

Řádkové styly nedoporučuji z nejrůznějších důvodů používat, protože:

- při tvorbě nového rozvržení webu musíte řádkové styly pracně odstraňovat z existujícího zdrojového kódu.
- lidé se zrakovým postižením si v internetových prohlížečích vytvářejí svoje vlastní styly, jež pak používají na všechny weby. Protože řádkové styly mají nejvyšší prioritu, nelze na internetových stránkách s řádkovými styly vlastní styly vůbec použít.
- řádkové styly zbytečně zvyšují množství zdrojového kódu.

Na druhou stranu se nenechte kýmkoliv a ani výše uvedenými tvrzeními ovlivnit natolik, abyste od této chvíle řádkové styly začali zatracovat. Existují totiž situace, kdy je použití řádkových stylů zcela na místě. Tak například můžete chtít uvnitř modulu vložit obrázek na pozadí a poté jej čas od času obměňovat. V tomto případě nemá smysl upravovat soubory CSS. Bylo by to daleko složitější než použít řádkové styly.

Když to tedy shrneme: nejlepší místo pro uložení stylů je externí soubor v hlavičce internetové stránky. Jedná se o čistě textový externí soubor, který můžete upravovat v jakémkoliv textovém editoru, například v Poznámkovém bloku.

Já sama se řadím ke spíše konzervativním osobám a všechny svoje soubory upravuji v programu Phase 5 od Ulliho Meybohma.

Selektory CSS

Slovo selektor při prvním setkání vyvolává dojem něčeho komplikovaného a zmateného. Ale nebojte se, není tomu tak.

Kaskádový styl musí vědět, kde přesně má platit, a k tomu potřebuje selektory. Mezi takové selektory patří kupříkladu elementy `p`, `h1–h6`, `div` a `span`. Elementům samozřejmě můžete přiřazovat i třídy a identifikátotry, které pak rovněž fungují jako selektory, ale o tom až později.

Selektory elementu

Ze všeho nejdříve vám na velmi jednoduchém příkladu objasním syntaxi.

Ve výše uvedeném příkladu s řádkovými styly jsme přiřadili odstavci červené písmo. Tento příkaz nyní můžete použít pro všechny elementy `p` v dokumentu:

```
p {color:#cc0000;}
```

Značka `p` zde funguje jako selektor. Za ním jsou uvedeny složené závorky, v nichž je uvedena vlastnost oddělená od své hodnoty dvojtečkou. Konec příkazu je vyznačen středníkem. (V tomto případě bychom mohli středník i vynechat, protože se jedná o poslední – nebo lépe řečeno jediný – příkaz v této deklaraci.)

Pokud bychom však chtěli elementu přiřadit více deklarací, pak se znak středníku uvést musí:

```
p
{
 color:#cc0000;
 background:#eee
 }
```

Pokud byste chtěli tento příkaz přiřadit více elementům, pak je můžete seskupit. K seskupení se používá znak čárky. Pozor! Za posledním selektorem již čárka být **nesmí**.

```
p, h1, h2
{
 color:#cc0000;
 background:#eee
 }
```

Třídy CSS pro vícenásobné použití stylů

Pokud byste potřebovali červenou barvu písma použít jen u některých odstavců, pak je nejjednodušším řešením použití třídy.

Používání tříd má však smysl pouze tehdy, pokud je plánujete použít vícekrát.

V našem zdrojovém kódu bude definice třídy vypadat například takto:

```
<p class="attention"> Odstavec s červeným písmem.</p>
```

Tady je příslušný kaskádový styl:

```
.attention {color:#cc0000;}
```

Všimněte si tečky před výrazem `attention`. Tato tečka uvnitř kaskádového stylu odkazuje na třídu.

Třídy je možné použít u nejrůznějších elementů HTML. Takto byste třeba mohli přiřadit červenou barvu písma nadpisu první třídy.

```
<h1 class="attention"> Toto je nadpis </h1>
```

Seskupování tříd

Třídy je rovněž možné seskupovat a toto seskupování představuje velmi flexibilní řešení. Nyní vám chci na příkladu standardního výstupu ze systému Joomla! ukázat, že seskupování tříd má daleko větší uplatnění v praxi, než se může zdát, když se budeme pohybovat na teoretické rovině. Vzhledem k tomu, že v tuto chvíli ještě nevíte o struktuře šablon systému Joomla! vůbec nic, prosím vás, abyste se nevyděsili! Jen čtěte dál a neutíkejte k počítači, abyste se podívali na šablonu.

Jak již asi víte, systém Joomla! vám umožňuje nastavit si pomocí parametrů v zobrazeních blogu počet sloupců, které se mají zobrazovat. Ve variantě s tabulkami v systému Joomla! 1.5 to není žádný problém, protože se tabulka zvětší o jeden sloupec v okamžiku, kdy zvýšíte hodnoty v parametrech o jedničku. Nic se nepokazí, vaše stránka se jednoduše odpovídajícím způsobem zvětší.

Nyní si zkusíme dosáhnout stejného zobrazení pomocí jednotlivých elementů `div`. Elementy `div` patří mezi významově prázdné. Tyto elementy se dají na monitor uspořádat tak, že se vytvoří viditelná mřížka internetové stránky. Následně pak pojmou libovolný počet dalších elementů, které se navíc mohou takřka neomezeně vnořovat:

```
<div><p>Odstavec</p></div>
```

Pokud má mít internetová stránka více sloupců, pak musí mít elementy nastavenu určitou šířku, aby byly správně zarovnány. Jinými slovy to znamená, že budete potřebovat alespoň jednu třídu, která bude obsahovat údaj o šířce sloupce. Pokud by měl navíc mít první sloupec jinou barvu než druhý, pak budete potřebovat další třídu. Jedné oblasti můžete nyní přiřadit dvě třídy.

Šířka sloupce se nastavuje pomocí vlastnosti `width`, které se jako hodnota zadává číslo v pixelech, popřípadě se použije relativní hodnota v jednotce `em` nebo `%`. Další informace najdete v kapitole *3.7* s názvem *Pozicování a box model*.

```
<div class="Sirka Barva"> <p> Odstavec </p> </div>
<div class="Sirka Barva2"> <p> Odstavec </p> </div>
```

Příslušný kaskádový styl by pak vypadal následovně:

```
.Sirka{width:200px}
.Barva { background:#cccc99} // světle šedá
.Barva2 {background:#999900} // zelená
```

Obrázek 3.1: Takto vypadá výsledek

Internetové stránce s jedním sloupcem odpovídá následující kód:

```
<div class="item column-1" ></div>
```

U internetové stránky se dvěma sloupci má zdrojový kód tuto podobu:

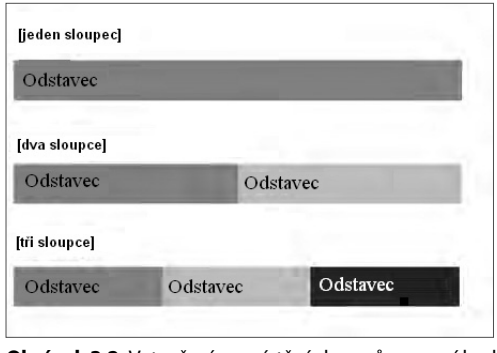
```
<div class="item column-1" ></div>
<div class="item column-2" ></div>
```

Internetová stránka se třemi sloupci má následující zdrojový kód:

```
<div class="item column-1" ></div>
<div class="item column-2" ></div>
<div class="item column-3" ></div>
```

Nyní byste mohli třídě `item` přidat obecná formátování platná pro všechny sloupce. Zkuste třeba typ písma. Pomocí tříd column-1, column-2 a column-3 se dá nastavit šířka a umístění různých sloupců.

Výsledek by mohl vypadat třeba takto:



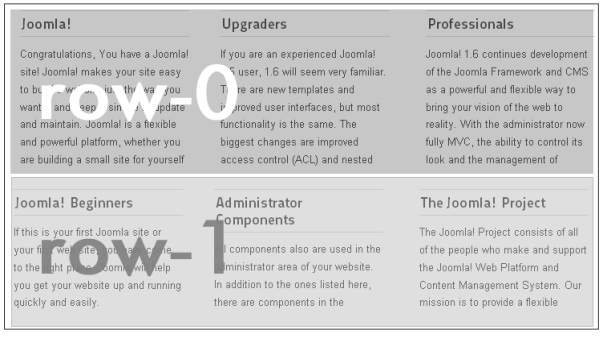
Obrázek 3.2: Vytvoření a umístění sloupců pomocí kaskádových stylů

Kromě toho můžete vytvářet i každý řádek zvlášť.

Předpokládejme šest položek, které chceme umístit do tří sloupců. Budeme tedy potřebovat dva řádky. Zdrojový kód by pak v systému Joomla! vypadal takto:

```
<div class="items-row cols-3 row-0">
<div class="item column-1" ></div>
<div class="item column-2" ></div>
<div class="item column-3" ></div>
</div>
```

```
<div class="items-row cols-3 row-1">
<div class="item column-1" ></div>
<div class="item column-2" ></div>
<div class="item column-3" ></div>
</div>
```



Obrázek 3.3: Řádky a sloupce

Systém Joomla! každou položku na řádku uzavře vždy jedním vlastním elementem `div`. Tento element má sám zase svoje další vlastní třídy. Jedná se o univerzální třídu, která je u každého řádku stejná, a to `items-row`. Poté je zde další třída, která udává počet sloupců, a konečně třída, která informuje o právě vypisovaném řádku:

- `.items-row` je univerzální třída, která je u každého řádku stejná.
- `.cols-3` udává počet sloupců.
- `.row-0 -row-xx` udává počet řádků.

Kombinací těchto tříd pak vzniká spousta možností uspořádání.

Podle toho, jaké zobrazení sloupců si nastavíte, můžete dále provádět další vlastní formátování. Fantazii se přitom meze nekladou.

Další obrovskou výhodou je i to, že nejste omezeni počtem zobrazitelných sloupců. Klidně si tak můžete ve správcovské části zvolit třeba deset sloupců, ovšem pak je musíte pomocí CSS odpovídajícím způsobem naformátovat.

Selektory identifikátoru

Nejprve to nejdůležitější: Samotný identifikátor se může v každém souboru vyskytnout pouze jednou. Tato skutečnost dostatečně objasňuje, proč se identifikátor v kostře HTML často používá pro umisťování jednotlivých částí stránky.

Kód HTML pak vypadá následovně:

```
<div id="header">
  <p>Veškerý obsah hlavičky</p>
</div>
```

Uvnitř souboru CSS pak k tomuto kódu můžete přistupovat takto:

```
#header
{
background:#000;
color:#fff
}
```

Před selektor identifikátoru se v CSS uvádí znak #.

Umisťování jednotlivých částí stránky patří bezpochyby do té profesionálnější úrovně práce s CSS a kvůli používání různých internetových prohlížečů není vždy snadné ho provést.

Kontextové selektory

Kontextové selektory patří mezi velmi praktické a kód šetřící způsoby pro formátování jednotlivých částí stránky pomocí kaskádových stylů.

Předpokládejme, že máme v části internetové stránky vyhrazené pro hlavičku nějaký seznam a chtěli bychom jej formátovat.

Kód HTML pak bude vypadat nějak takto:

```
<div id="header">
  <ul>
    <li>1. položka seznamu </li>
    <li>2. položka seznamu </li>
  </ul>
</div>
```

K tomuto kódu lze pak přistoupit takto:

```
#header ul li {color:#cc0000;}
```

V tomto případě budou všechny položky seznamu uvnitř hlavičky napsány červeně. Elegantní, že?

Selektory potomka

Selektory popisované v této části jsou velmi praktické, ale mají tu nevýhodu, že je nepodporuje internetový prohlížeč Internet Explorer 6.

```
body>p
```

Tento kód provede formátování všech `p`, které jsou přímým potomkem `body`.

Přímým potomkem se zde rozumí takový, mezi nímž a rodičem neleží žádný další element. V následujícím příkladu by selektor nenašel žádné využití, protože odstavec není přímým potomkem elementu `body`, ale podřízeného elementu `div`.