PŘÍPADOVÁ STUDIE 1

Více než čmáranice

Simon Collison

V celé této knize Andy podrobně popisuje některé CSS metody, které se těžko implementují a mnohdy bývají zcela nepochopeny nebo špatně použity. Je jasné, že jako návrháři máme širokou paletu možností, ale zároveň je jasné, že v některých případech je použití úplně prvního řešení, jež vás napadne, nepříliš optimálním postupem. Vědom si tohoto faktu jsem se poradil s Cameronem a nakonec jsme rozhodli o zabudování velkého množství těchto CSS metod do dvou experimentálních layoutů, abychom zjistili klady a zápory obou přístupů a ilustrovali jejich použití na dvou funkčních, přístupných a současně standardům vyhovujících webových stránkách.

Mnozí z nás si chceme ulehčit práci. Chceme mít úplnou kontrolu nad našimi layouty a dosáhnout maximálního efektu při minimálním značkování. Rozhodně pravdivá je skutečnost, že tato schopnost vychází zejména z trpělivosti a praxe, nicméně přidáním několika dodatečných klíčových hacků do vašeho XHTML si ponecháte volnost pracovat výhradně s vaším CSS kódem a vytvořit z použitých XHTML prvků na pohled úžasný grafický bonbónek.

V této případové studii se naučíte:

- Ovládat oblast pro obsah pomocí selektorů potomka (descendant selectors).
- Vytvořit plovoucí sloupce.
- Zvýraznit aktuální stránku na základě třídy v prvku `body`.
- Vytvořit obsah pro jednotlivé sloupce.
- Používat průhledné uživatelské rohy a orámování.
- Kombinovat třídy pro cílené zaměřování se na prvky.
- Používat třídy pro obrázky.
- Pracovat s odkazy.
- Vytvářet plovoucí stíny pro různé prvky na stránce.

O této případové studii

Tato případová studie vám ukáže, jak si poradit s vytvářením jednoduchého sémantického značkování a jak v praxi používat existující CSS techniky co nejsnadnějším a nejefektivnějším způsobem. Kód vaší stránky už nebude obsahovat různé zbytečné prvky `div`, přičemž se omezí i nadbytečný balast. CSS je totiž chytřejší, než by se mohlo na první pohled zdát. Ostatně – tento fakt byl autorem knihy dostatečně demonstrován ve všech předcházejících kapitolách. Příklady v této případové studii jsou založeny na různých užitečných schopnostech CSS jako jsou selektory potomka, selektory s atributy, dědičnosti a mnoha dalších, které vykonají všechnu těžkou práci, přičemž XHTML kód zůstane z velké části přehledný a čistý.

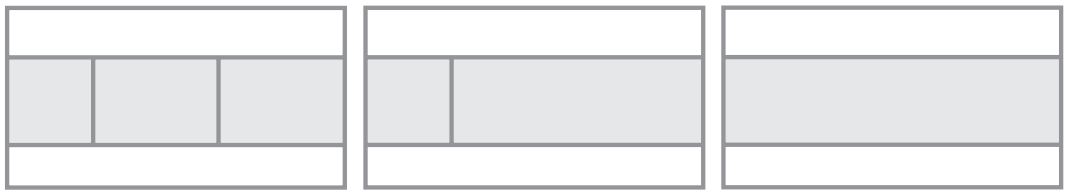
V této studii získáte klíč ke stránkám More Than Doodles ("Více než čmáranice", viz obrázek 1), což je kompletně fiktivní web, který na svých stránkách představuje skutečně ilustrátory. Nápad vytvořit zpravodajský server zaměřený na ilustrace a digitální umění se zdá být vhodný vzhledem k tomu, že by se nenašel ani jeden dobrodružný ilustrátor, který by chtěl provozovat takové stránky bez svobodného používání obrázků a perfektně vypadajícího vzhledu s několika drobnými vylepšeními. Pro tyto stránky bude potřeba použít hodně obrázků, ale takovým způsobem, aby stránka nebyla přeplněna a nebyla z datového hlediska příliš velká. Takže – nyní je ten správný čas, jak sebrat techniky popisované v této knize a začít je používat v praxi.

Obrázek 1. *Hlavní stránka More Than Doodles (Více než čmáranice).*

Tvorba nového layoutu webu přestává být legrací, když hlavním smyslem této studie je především rychlé pochopení spousty metod a postupů. Díky zajímavým technikách popisovaným v této knize ovšem zjistíte, jak jednoduché je pozvednout layout na vyšší úroveň, přičemž doufám, že s vytvořeným layoutem budete ve svém volném čase dále experimentovat.

Ovládání oblasti pro obsah pomocí selektoru potomka

Naši práci na stránkách začneme vytvořením oblasti pro obsah, která bude umístěna mezi hlavní horizontální navigací umístěnou v hlavičce a patičkou v dolní části stránky, viz obrázek č. 2. Řečeno jinými slovy – obsahová oblast je oblast, kde bude umístěn hlavní obsah stránky. Tuto oblast můžete navrhnout třemi způsoby – buď bude jednosloupcová, dvousloupcová (úzký sloupec vlevo a širší sloupec napravo), nebo třísloupcová (úzký sloupeček vlevo a dva stejně široké sloupce vpravo). V závislosti na tom, co budou jednotlivé stránky webu obsahovat, máte možnost dynamicky ovládat zobrazení jednotlivých sloupců pomocí selektoru potomka.



Obrázek 2. *Obsahová oblast je zvýrazněna šedou barvou. Možné rozvržení sloupců.*

Selektory potomka vám nabízejí naprostou kontrolu nad vaším layoutem. Pro zopakování – selektor `h3 {color: #000}` vykreslí všechny nadpisy třetí úrovně v dokumentu černou barvou. To je snadné. Dále si řekněme, že v postranním pruhu máte nějaký nadpis třetí úrovně, u kterého chcete, aby nebyl zobrazen černě, ale například červeně. To je taky snadné – jednoduše specifikujte selektor `h3` jako potomka postranního pruhu, např. `#sidebar h3 {color: #FF0000}`. Takže v tuto chvíli máte v jednom pravidle nadefinovány dva selektory, které jsou odděleny kombinátorem (v tomto případě jako kombinátor slouží mezera), což vám umožní se zaměřit na specifické části vašeho layoutu. Layout webu More Than Doodles takových selektorů potomků používá mnoho.

Takže – proč nepoužít tuto metodu pro prvek `body` v každé jednotlivé stránce webu? Tím, že tomuto prvku přiřadíte nějakou třídu a identifikátor, který bude odlišný pro jednotlivé XHTML stránky, získáte možnost jednoduše ovládat velké množství selektorů ve vašem kódu CSS. Jednoduše řešeno – pomocí stylu nebude problém ostylovat jednotlivé stránky webu různým způsobem.

XHTML

Dále se budeme zabývat XHTML kódem pro vytvoření oblasti pro obsah. Každému ze tří sloupců specifikujeme jedinečný identifikátor. Struktura kódu pro sloupce s obsahem vypadá takto:

```
<div id="primaryContent">
```

```
  obsah
</div>
<div id="secondaryContent">
  obsah
</div>
<div id="sideContent">
  obsah
</div>
```

Pokud byste chtěli, aby nějaká konkrétní stránka našeho ukázkového webu měla menší počet sloupců, měli byste z XHTML kódu dané stránky odstranit příslušné prvky. Tím se sníží datová velikost stránky a urychlí se její načítání, nehledě na to, že nebude docházet ke zmatení vyhledávacích strojů. Nicméně – pro účely této případové studie nebudeme sloupce ze zdrojového kódu odstraňovat, protože je budeme zobrazovat nebo schovávat v závislosti na atributech použitých v prvku `body`.

Poznámka ohledně použité konvence pro názvy sloupců

Pro účely této případové studie používám pro jednotlivé sloupce názvy, které výstižně popisují jejich význam. Kdo dnes ovšem může s naprostou jistotou říci, že sloupec `#primaryContent` (sloupec pro primární obsah) bude zobrazovat hlavní obsah i v budoucnosti? Nikdo. Po nějaké době jej třeba začnete používat pro zobrazení sekundárního obsahu, přičemž původní název sloupce pak může vyvolávat zbytečné zmatky (například se můžete sami sebe ptát, proč sloupec pojmenovaný jako `#primaryContent` zobrazuje druhotný obsah stránky). Pro účely této konkrétní případové studie je to ovšem jedno, nehledě na to, že při použití těchto pojmů budete schopni si lépe představit strukturu webové stránky.

Třísloupcové rozvržení

Přeskočme rovnou k třísloupcovému rozvržení, které bude použito na hlavní stránce. Když prvku `body` přiřadíte identifikátor `threeColLayout`, prostřednictvím tří CSS pravidel popisovaných dále dosáhnete toho, že úvodní stránka našeho webu se bude skládat ze tří sloupců – z tenké postranní lišty vlevo a dvou stejně širokých sloupců napravo od ní.

```
<body id="threeColLayout">
```

Zde je definice stylu pro primární obsah webu:

```
#threeColLayout #primaryContent {
float:left;
width:270px;
margin: 0 0 20px 195px;
}
```

Identifikátor `threeColLayout`, který je použitý v prvku `body`, poskytuje důležitý záchytný bod. Všimněte si, že v definici stylu pro primání obsah je použit selektor potomka – `#threeColLayout` je následován `#primaryContent`. To v praxi znamená, že toto pravidlo pro primární sloupec se použije pouze tehdy, pokud prvek s identifikátorem `#primaryContent` bude potomkem prvku s `#threeColLayout`. Řečeno jiným slovy – tento styl se aplikuje ve stránce pouze tehdy, pokud prvek `body` bude mít v atributu `id` hodnotu `threeColLayout`.

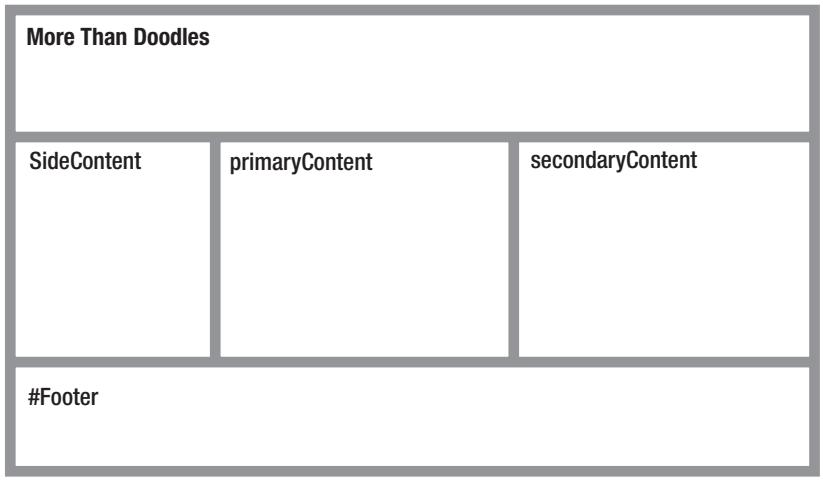
Pravý sloupec, který je určený pro sekundární obsah, má následující definici (tento sloupec se používá pouze při třísloupcovém layoutu):

```
#secondaryContent {
float:left;
width:270px;
margin: 0 0 20px 15px;
}
```

A nakonec tu máme definici levé postranní lišty.

```
#sideContent {
float:left;
width:180px;
margin: 0 0 20px -750px;
}
```

To, proč jsou tyto sloupce plovoucí, si popíšeme v této případové studii později. Povšimněte si, že oba sloupce pro obsah mají nastavenou shodnou šířku 270 pixelů. Společně s postranním sloupcem o šířce 180 pixelů tak vytváří třísloupcové rozvržení, které je zobrazeno na obrázku č. 3.



Obrázek 3. *Třísloupcové rozvržení.*

Dvousloupcové rozvržení

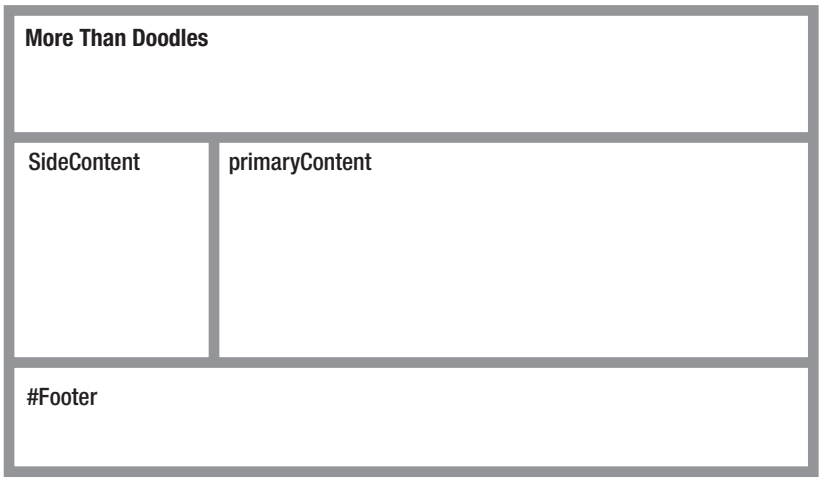
Dvousloupcový layout je použit na stránce s galerií. Ve zdrojovém kódu stránky pro galerii nastavte atribut `id` v prvku `body` na hodnotu `twoColLayout`:

```
<body id="twoColLayout">
```

Do stylového předpisu pak přidejte následující pravidlo:

```
#twoColLayout #primaryContent {
width:555px;
float:left;
margin: 0 0 20px 195px;
}
```

Jak je dobře vidět, je opět použit selektor potomka. Protože v této stránce je prvek s identifikátorem `primaryContent` potomkem prvku `body` s `twoColLayout` (a nikoliv potomkem prvku `body` s `threeColLayout`, jak tomu bylo v případě úvodní stránky), pro primární sloupec se nepoužije pravidlo `#threeColLayout`, ale pravidlo `#twoColLayout`. V důsledku tohoto bude mít sloupec pro hlavní obsah šířku 555 pixelů místo původních 270 pixelů, a to prosím bez jakýchkoliv úprav ve struktuře sloupců ve zdrojovém kódu stránky (viz obrázek 4).



Obrázek 4. *Dvousloupcové rozvržení.*

Jednosloupcové rozvržení

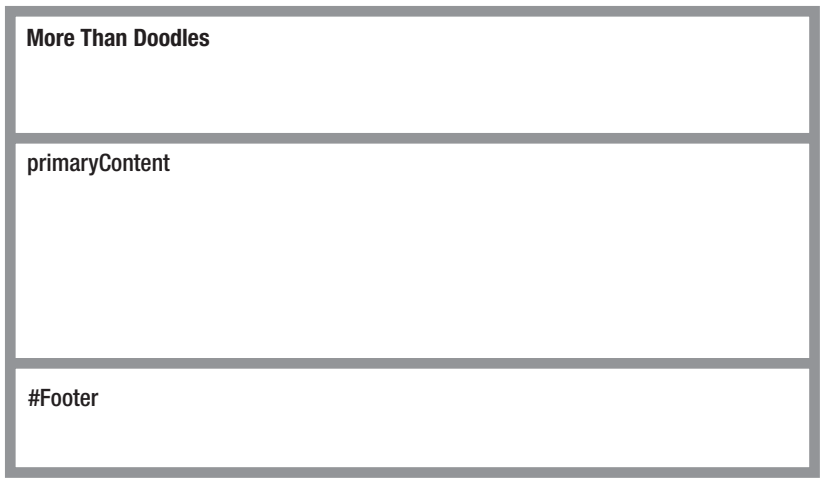
Jednosloupcové rozvržení je použito na stránce s kontakty. A stejně jako v minulých případech, i zde změníme hodnotu atributu `id` v prvku `body`.

```
<body id="oneColLayout">
```

V definici stylu pro hlavní sloupec už nepoužijeme selektor potomka – není to potřeba. Takže pro hlavní sloupec ve stránce s kontakty se aplikuje následující definice stylu:

```
#primaryContent {
width:750px;
margin: 0 0 20px 0;
background: #FFF;
}
```

Tato definice vytvoří jeden jediný sloupec pro hlavní obsah, který bude mít plnou šířku layoutu, jak je to názorně ukázáno na obrázku 5.



Obrázek 5. *Jednosloupcové rozvržení.*

Odstranění nežádoucích sloupců

Pro stránku s galerií chcete použít dvousloupcové rozvržení, nicméně ve zdrojovém kódu stránky je stále ponechán kontejner s identifikátorem `secondaryContent` pro vedlejší obsah. Definice stylu pro tuto stránku nedává tomuto sloupci žádnou jinou možnost, než se přesunout ze svého původního umístění pod ostatní sloupce. Prvek `div` s identifikátorem `secondaryContent` by měl být v ideálním případě ze zdrojového kódu stránky úplně odstraněn (už jenom z hlediska velikosti webové stránky), nicméně pro účely naší případové studie jsme ho tam ponechali, takže je potřeba jej ve stránce zneviditelnit. To je opět zařízeno pomocí CSS s využitím selektoru potomka:

```
#twoColLayout #secondaryContent {
display: none; }
```

Díky tomuto pravidlu bude na stránce s dvousloupcovým rozvržením kompletně ukryt sloupec, který má přiřazený identifikátor `secondaryContent`.

Tento postup můžeme použít i v případě layoutu, který obsahuje pouze jeden hlavní sloupec. Abychom ukryli dva nepoužívané sloupce, musíme v jednom pravidle použít dva selektory potomka, čímž jim přiřadíme stejnou hodnotu. Takže – pomocí následujícího pravidla s deklarací `display: none` ukryjeme v jednosloupcovém layoutu nejenom levý postranní sloupec, ale také pravý sloupec pro sekundární obsah.

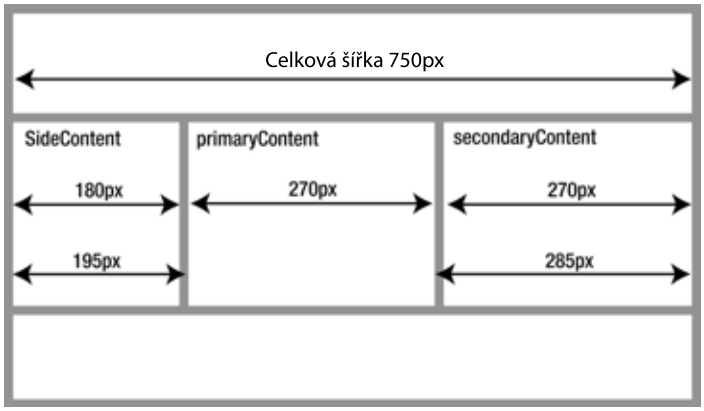
```
#oneColLayout #secondaryContent, #oneColLayout #sideContent
{ display: none; }
```

Deklaraci `display:none` používejte s rozmyslem – rozhodně nemá smysl ji používat pro prvky, které v dané stránce nemají co dělat. Takové prvky byste měli z vašeho kódu rovnou odstranit. Různé webové vyhledávače, jako je například Google, se rovněž mohou zajímat, proč potřebujete před uživatelem skrývat nějakou část kódu a mohou vás kvůli tomu penalizovat – kvůli domněnce, že obsah skrytý před uživatelem má sloužit pouze pro získání lepší pozice ve výsledcích vyhledávání.

Plovoucí sloupce

Někde v této případové studii jsem zmínil, že sloupce pro obsah jsou plovoucí. Ptáte se proč? Jednoduše proto, že existuje několik velmi dobrých důvodů, proč v našem ukázkovém layoutu použít plovoucí sloupce. Například – použití záporných okrajů v CSS vám umožní specifikovat, kde v designu se mají objevit naše tři sloupce bez toho, abyste museli ustupovat od sémanticky navrženého XHTML kódu stránky. Podívejte se například na dvousloupcový layout. Z vizuálního hlediska to vypadá tak, že sloupec pro primární obsah je umístěn uprostřed designu a je umístěn napravo od méně důležitějšího postranního sloupce. Když ovšem v prohlížeči vypnete CSS, obsah primárního sloupce se vzhledem ke svému umístění ve zdrojovém kódu stránky zobrazí nad obsahem postranního sloupce.

Není ale všechno tohle pozicování se zápornými okraji tak trochu komplikované? Bohužel je – ale pokud si na začátku všechno dobře spočítáte (viz obrázek č. 6), všechno zapadne na své místo.



Obrázek 6. *Výpočet rozměrů.*

Výpočty

Nyní zapněte kalkulačku. Protože naše prvky `div`, které vytvářejí jednotlivé sloupce, mají specifikovány vlastní okraje, není potřeba jim definovat nějakou výplň. Díky tomu nemusíme řešit problémy vznikající kvůli tomu, že některé webové prohlížeče používají model pro výpočet rozměrů boxu, jenž je v rozporu se specifikací CSS.

Takže – v našem případě budeme hlavně pracovat se šířkou omezujícího kontejneru a se šířkou a okraji jednotlivých prvků `div`, které vytvářejí sloupce.

Nejprve se podívejme na omezující prvek `div` s identifikátorem `wrapper`.

```
#wrapper {
width:750px;
margin:0 auto;
padding: 0 10px 10px 10px;
background-color: #D7D493;
}
```

Toto CSS zajistí, že náš ukázkový layout bude široký přesně 750 pixelů. Vlevo a vpravo je sice přidána výplň o velikosti 10 pixelů, nicméně toto neovlivní šířku prostoru dostupného pro jednotlivé sloupce. Nyní se opět podívejme na definice jednotlivých sloupců v případě třísloupcového layoutu:

```
#threeColLayout #primaryContent {
float:left;
width:270px;
margin: 0 0 20px 195px;
}
#secondaryContent{
float:left;
width:270px;
margin: 0 0 20px 15px;
}
#sideContent{
float:left;
width:180px;
margin: 0 0 20px -750px;
}
```

Co tady máme zajímavého? Šířky jednotlivých sloupců. Když je pečlivě sečteme (270 + 270 + 180), dostaneme hodnotu 720 pixelů. To znamená, že nám zbývá 30 pixelů, které využijeme k tomu, abychom od sebe odsadili jednotlivé sloupce. A protože máme celkem tři sloupce, budeme potřebovat dvě mezery mezi nimi, abychom je od sebe odsadili, takže každá mezera bude velká 15 pixelů. Tím dosáhneme toho, že všechny tři sloupce, včetně mezer mezi nimi, zaberou oněch 750 pixelů.

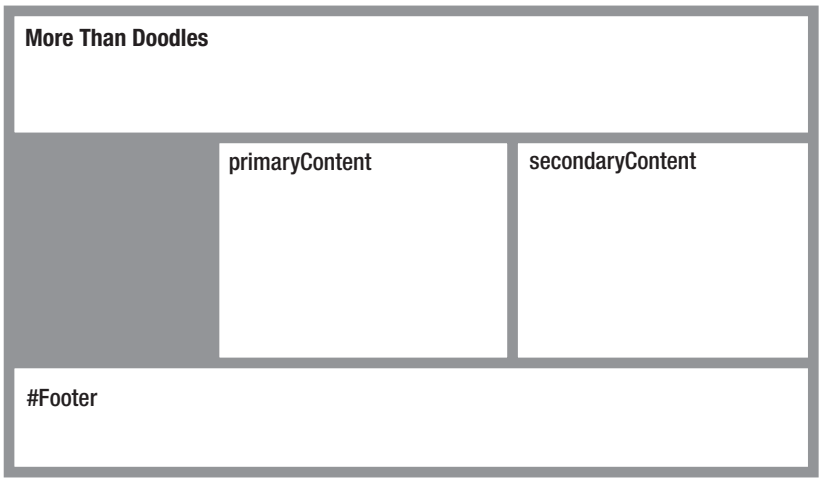
Když se dobře podíváte na výše uvedené CSS, zjistíte, že je v něm specifikován pouze jeden okraj o konkrétní velikosti 15 pixelů – jedná se o okraj, co je uveden pro `#secondaryContent`, který zajišťuje 15 pixelů velké odsazení od sloupce pro primární obsah. A kdepak máme to druhé odsazení? Když se podíváte na obrázek č. 6, zjistíte, že levý postranní sloupec je široký přesně 180 pixelů. Ve stylu pro sloupec `primaryContent` je ovšem nastaven levý okraj o velikosti 195 pixelů. A to je přesně ono. Tento okraj hlavního sloupce vytváří ve stránce prostor pro 180 pixelů široký postranní sloupec a současně prostor pro 15 pixelů velké odsazení mezi těmito sloupci. Elegantní.

Umístění sloupců na správné místo

Šířky jednotlivých sloupců máme vyřešeny. Ale to, co ještě nemáme vyřešeno, je pořadí těchto sloupců. Z vizuálního hlediska je chceme mít umístěny ve stránce v pořadí postranní sloupec, primární sloupec a sekundární sloupec. V (X)HTML kódu ovšem musí jejich pořadí reflektovat jejich důležitost, takže pořadí sloupců v kódu bude odlišné od pořadí, ve kterém se zobrazují v prohlížeči. V kódu bude první primární sloupec, pak sekundární a nakonec postranní sloupec.

Ve stylu musíte pro každý sloupec specifikovat deklaraci `float: left`, aby byly sloupce srovnané pěkně zleva doprava. Kdybyste tuto deklaraci nepoužili, jednotlivé sloupce by se seřadily za sebe, s tím, že kvůli nastaveným okrajům by byly různě odsazeny od omezujícího prvku `div`.

Kdybyste nyní ze zdrojového kódu odstranili značkování pro postranní sloupec, nebo jej pomocí vlastnosti `display` a hodnoty `none` skryli, primární sloupec bude stále z levé strany odsazen o 195 pixelů, přičemž sekundární sloupec bude stále umístěn napravo od primárního sloupce, viz obrázek 7. Pokud byste v CSS kódu pro primární sloupec nastavili nulové odsazení zleva, oba sloupce se posunou vlevo, k levé hraně omezujícího prvku `div`.

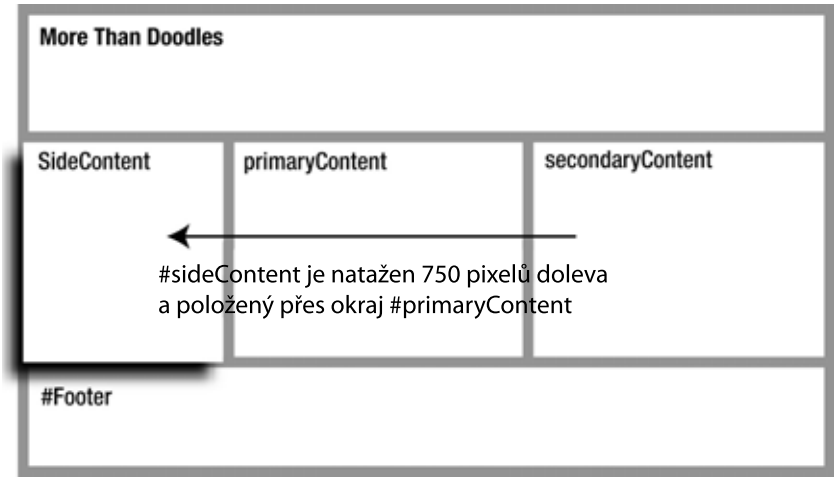


Obrázek 7. *Třísloupcové rozvržení s ukrytým postranním sloupcem.*

Možná vám vrtá hlavou, jak je možné, že postranní sloupec, který je ve zdrojovém kódu stránky specifikován až za oběma zbývajícími sloupci, je ve stránce zobrazen jako první, tzn. před sloup-

cem pro primární obsah a sloupcem pro vedlejší obsah? Je to velmi jednoduché – tím, že primární sloupec má nastaven levý okraj o velikosti 195 pixelů, jsme ve stránce vytvořili volné místo pro postranní sloupec (včetně místa pro 15 pixelů velké odsazení). A tento postranní sloupec jsme do toho volného místa posunuli pomocí záporného okraje o velikosti 750 pixelů (viz obrázek 8).

Abych to vysvětlil ještě lépe. Kdyby všechny tři sloupce neměly nastaveny žádné vodorovné okraje, zobrazovaly by se na stránce v pořadí, které by bylo shodné s jejich pořadím v kódu stránky. První by byl primární sloupec, pak sekundární sloupec a nakonec postranní sloupec. Ale díky tomu, že nastavíte postrannímu sloupci záporný levý okraj o velikosti 750 pixelů, čímž jej posunete vlevo a současně pro něj uděláte prostor odsazením primárního sloupce zleva o 195 pixelů, bude se vše zobrazovat podle našich představ.



Obrázek 8. *Díky zápornému okraji je postranní sloupec posunut úplně vlevo.*

Protože selektor `#sideContent` má tento záporný okraj nastaven vždy, ať už se jedná o potomka nebo ne, bude tento postup bezproblémově funkční jak ve třísloupcovém, tak i ve dvousloupcovém layoutu.

S ohledem na tyto skutečnosti byste si tedy měli uvědomit, že opatrným nastavováním levého nebo pravého okraje jednotlivých sloupců je možné dosáhnout libovolného seřazení sloupců na stránce, bez ohledu na to, jak jsou uspořádány ve zdrojovém kódu stránky.

Zvýraznění aktuální stránky pomocí třídy v body

Na předchozích stránkách jste se dozvěděli o výhodách použití atributu `id` v prvku `body`. V naší případové studii toto používáme ke změně počtu sloupců na vybraných stránkách. Nyní je čas přidat do prvku `body` atribut `class`, který využijeme k tomu, aby se v hlavním menu stránky automaticky zvýrazňovala aktuálně zobrazená stránka, viz obrázek 9.