Kapitola 4

Pokročilá optimalizace HTML

Zkracovat lze pouze kód HTML, který je využit pro zobrazení stránky v prohlížeči. Pro úplnou optimalizaci HTML se musíte do kódu více ponořit a změnit jej. Tato kapitola popisuje, jak vyladit tabulky, zoptimalizovat formuláře, zkrátit adresy v odkazech a zkomprimovat HTML – to vše pro dosažení maximální rychlosti stažení a zobrazení stránky. Poslední dvě techniky vyžadují zvláštní kroky na serveru, které vysvětlím i v této kapitole, ale především v kapitole 17, Serverově orientované techniky a v kapitole 18, Komprimace webu.

Tipy na optimalizaci tabulek

Je škoda, že tabulky přišly před CSS. Mnoho designérů, kteří navrhují design stránek s přesností na jeden pixel, bývá často rozčarováno, když zjistí, jak málo toho HTML nabízí pro ovlivnění spolehlivého umístění všech prvků na stránce. Někteří průkopníci, například David Siegel[1], který představil trik používající jedno-pixelový neviditelný GIF, zjistili, že k hrubému rozmístění prvků na stránce je možné použít tabulky. Následovala exploze používání tabulek.

Tabulky byly původně zamýšleny pro zobrazování tabelovaných dat a jejich nadměrné používání pro design doslova znečistilo web na miliardách stránek zbytečnými formátovacími strukturami. S příchodem moderních prohlížečů uznávajících standardy je možné tabulky nahradit lepší alternativou kterou je CSS2. Ale protože je tato část věnována tabulkám, předpokládám, že bych měl hovořit o technikách, které urychlí jejich zobrazení.

Složité tabulky = pomalé zobrazení

Používání tabulek pro zalamování složitých stránek mnohdy vede k pohromě. Obsah nacpaný do tří nebo čtyř sloupců ve struktuře, která byla původně vyvinuta pro uchovávání vědeckých dat, je velmi vzdálen původní myšlence webu definované organizací W3C. Designéři pracující s HTML editory ve WYSIWYG režimu obvykle zalomí stránku do jedné velké tabulky, do které vloží celý obsah. Často je tento obsah rozdělen na další tabulky, které mohou obsahovat opět velké množství dalších tabulek. Problém spočívá v tom, že výpočty velikostí tabulek jsou dynamické a navzájem provázané.

Veškeré takové "spletitosti" musí být prohlížečem "rozmotány" a zpracovány, což je jeden z důvodů, proč je tak mnoho webových stránek zobrazováno pomalu, dokonce i s použitím velmi rychlého připojení k internetu. Čím je tabulková struktura složitější, tím je její zobrazování pomalejší, zejména při použití obrázků, které ji vyplňují nebo určují její rozměry. Každý z těchto obrázků generuje jeden požadavek HTTP, čímž výrazně zpomaluje zobrazení stránky.

Řešení, jak urychlit tabulky, spočívá v předávání pouze takového množství informací prohlížečům, které je nezbytně nutné k vytvoření struktury a zobrazení obsahu a také v redukci složitosti tabulky. Zkuste tabulky co nejvíce zjednodušit a "rozmotat". Pro naformátování tabulek použijte CSS, místo obrázků na pozadí použijte barvy a pokud je to možné, definujte tabulky s pevnými rozměry. Úplně nejlepším řešením je ale používat na zalamování obsahu prvky *div* a CSS2 místo tabulek. Ale tím se bude zabývat jiná kapitola (kapitola 8, Pokročilá optimalizace CSS). Začněme tedy s optimalizací tabulek.

Umožněte postupné zobrazování

Pokud mají tabulky dostatek informací, mohou se vykreslovat postupně podle toho, jak přicházejí jednotlivé řádky namísto toho, aby prohlížeč čekal na všechny údaje, než je začne zobrazovat. Pokud chcete, aby prohlížeč vykresloval tabulku během jednoho průchodu kódem, musíte specifikovat počet řádků v tabulce a jejich šířku. Tuto informaci předáte prostřednictvím prvků *colgroup* a *col*. Pokud mají některé sloupce relativní šířku, je potřeba také určit šířku tabulky. Pokud prohlížeč nedostane tyto přesné údaje, zpracovává tabulku pomocí dvou-průchodového algoritmu, přičemž se při prvním průchodu zjišťuje počet sloupců. Předáním přesných informací o struktuře tabulek můžete tedy urychlit jejich zobrazení.[2]

Řádky a sloupce v tabulkách lze seskupit, aby se daly formátovat jako skupiny. V prohlížečích vyhovujících standardům – například Internet Explorer 5 pro Mac, Internet Explorer 6, Netscape 6 a Opera 6 – získáte sloučením atributů ohromné úspory. Pomocí atributů *frame*, *rules* a *border* je pak možné řídit, jak se zobrazí externí rámec tabulky a jak se budou aplikovat interní pravidla.

Seskupování řádků

Řádky v tabulce mohou být použitím prvků *thead*, *tfoot* nebo *tbody* seskupeny do hlavičky, patičky nebo těla. Tyto skupiny se mohou formátovat odděleně a používat na tisk delších, vícestránkových tabulek, u kterých se má informace z hlavičky/patičky tisknout na každé stránce.

Například:

```
<table frame="border" rules="groups">
<thead>
<tr> ...informace v hlavičce...
<tfoot>
<tr> ...informace v patičce...
<tbody>
<tr> ...první řádek datového bloku jedna...
<tr> ...druhý řádek datového bloku jedna...
<tbody>
<tr> ...první řádek datového bloku dva...
<tr> ...druhý řádek datového bloku dva...
<tr> ...třetí řádek datového bloku dva...
</table>
```

Prvek *tfoot* se musí objevit ještě před prvkem *tbody*, takže prohlížeč může zpracovat patičku ještě před obdržením všech údajů o řádcích. Tato optimalizace používaná CALS[3] je vhodná ke zpracování velmi dlouhých tabulek. Umožňuje, aby byla patička zobrazena bez čekání na zpracování celé tabulky[4]. Ale pro naše účely je mnohem důležitější schopnost seskupování a specifikování vlastností sloupců.

Seskupování sloupců

Seskupování sloupců umožňuje strukturální zalamování tabulek a definovat styl sloupců pomocí CSS nebo atributu *rules* v prvku *table*.

Tabulka obsahuje jednoduchou implicitní skupinu sloupce (není uveden *colgroup*) nebo jednu a více přesně určených skupin sloupců oddělených prvkem *colgroup*.

Prvek *col* umožňuje přiřazení vlastností jednomu nebo více sloupcům. Pro vícenásobné sloupce se používá atribut *span*[5].

S použitím atributu *span* v prvku *colgroup* lze specifikovat na jednom místě atribut *width* pro více sloupců. Atribut *width* je také možné vložit do každého prvku *col*. Efektivnější je pochopitelně definice stejně širokých sloupců prostřednictvím *span*:

```
<colgroup span="10" width="60">
</colgroup>
```

Pro porovnání, zde je alternativa s *col*:

```
<colgroup>
<col width="60">
<col width="60">
...celkem deset prvků col...
</colgroup>
```

Zjištění počtu sloupců

V HTML 4.01 je možné vymezit počet sloupců v tabulce dvěma způsoby:

- použitím prvků *col* nebo *colgroup* (v tomto pořadí)
- pokud neexistují prvky sloupce, je vymezen počet sloupců vyžadovanými řádky

Počet sloupců v tabulce je shodný s řádkem s více buňkami používajícími *span*. Definováním údajů o sloupcích si mohou prohlížeče ušetřit průchod dovnitř tabulky ke zjištění, kolik sloupců je potřeba.

Zjištění šířky sloupců

Jakmile prohlížeč zjistí počet sloupců, počítá šířku každého sloupce. Šířku sloupce lze zadat třemi způsoby:

- **Pixely** – pevná šířka (například: *width="125"*), umožňuje postupné vykreslování tabulky.
- **Procenta** – procentuální šířka (například: *width="33%"*), je určena poměrným rozdělením vodorovného prostoru dostupného pro tabulku a pochopitelně také umožňuje postupné zobrazování tabulky.
- **Relativní** – relativní šířka (například: *width="2*"*), je založena na vodorovném prostoru vyžadovaného tabulkou. Pokud máte tabulku s pevnou šířkou, prohlížeč ji může stále vykreslovat postupně. Pokud však tabulka nebude mít pevnou šířku, prohlížeč musí obdržet veškeré údaje z tabulky před tím, než stanoví rozměry tabulky[6].

Postupné zobrazování tabulky tedy umožníte tím, že stanovíte počet a šířku sloupců. Pokud ne, prohlížeč bude muset počkat, než dorazí veškeré údaje nutné k rozdělení místa na stránce. Výpis 4.1 demonstruje příklad použití.

Výpis 4.1 – Specifikace šířky sloupců v tabulce

```
<table width="100%">
<colgroup>
<col width="33%">
<col width="67%">
</colgroup>
<tr>
```

```
<td colspan="2">
<p>horní navigační řádek (název webu a reklama)</p>
</td>
</tr>
<tr>
<td>
<p>levý navigační sloupec</p>
</td>
<td>
<p>oblast pro hlavní obsah stránky</p>
</td>
</tr>
<table>
```

Pevná nebo procentuální šířka umožňuje postupné zobrazování tabulky. Protože designéři nemají možnost ovlivnit nastavení tabulek a velikost písma na straně uživatele, spoléhání se na pevnou šířku tabulek je poměrně riskantní, pokud obsah nemá předem známou velikost.

Sloučení atributů v tabulce

Další místo se dá také ušetřit používáním výchozích hodnot a sloučením specifických atributů do atributů globálních. Pro definici zarovnání obsahu v buňkách použijte pouze jeden *<tr align="center">* místo mnoha výskytů *<td align='center'>*. Ještě lepší je ale použití skupin řádků (pomocí *thead* nebo *tbody*) pro definici celých bloků řádků. Výpis 4.2 uvádí příklad takového zápisu.

Výpis 4.2 – Sloučené atributy

```
<table width="100%" frame="border">
<colgroup>
        <col width="33%">
        <col width="67%">
</colgroup>
        <tr>
                <td colspan="2">
                        <p>horní navigační řádek (název webu a reklama)</p>
                </td>
        </tr>
        <tbody align="center">
        <tr>
                <td>
                        <p>levý navigační sloupec</p>
                </td>
                <td>
```

```
                    <p>oblast pro hlavní obsah stránky</p>
                </td>
        </tr>
        </tbody>
</table>
```

V HTML jsou uzavírací značky pro prvky skupin řádků a *col* nepovinné (*</thead>*, *</tbody>*, *</tfoot>* a *</col>*, nicméně jsou povinné v XHTML. Pozor, některé verze Netscape 4 chybují při uzavírání prvku *</colgroup>* pro *div* a mohou vést při vykreslování k nepředvídatelnému chování prohlížeče. Jednoduše řečeno, pro nastavení zarovnání jednoho nebo více sloupců můžete používat *colgroup* namísto individuálního zarovnávání každé buňky.

Například, používejte tento zápis pro zarovnání jednoho sloupce:

```
<colgroup align="center">
```

Nebo použijte tento zápis pro zarovnání skupiny tří sloupců:

```
<colgroup align="center" span="3">
```

Tato technika funguje v prohlížečích vyhovujícím standardům, přičemž ve starších prohlížečích je půvabně degradována (zarovnání doleva). Pochopitelně, kromě výše uvedeného, je možné pro nadefinování *col* a *colgroup* použít CSS (viz výpis 4.3).

Výpis 4.3 – Tabulka s využitím CSS

```
<style type="text/css">
<!--
        table#skel {width:100%;}
        col#left {width:33%;}
        col#right {width:67%;}
        tbody#main {text-align:center;}
-->
</style>
</head>
<body>
<table id="skel" frame="border">
<colgroup>
        <col id="left">
        <col id="right">
</colgroup>
        <tr>
                <td colspan="2">
                        <p>horní navigační řádek (název webu a reklama)</p>
                </td>
        </tr>
```

```
        <tbody id="main">
        <tr>
                <td>
                        <p>levý navigační sloupec</p>
                </td>
                <td>
                        <p>oblast pro hlavní obsah stránky</p>
                </td>
        </tr>
        </tbody>
</table>
```

Všechny tyto informace dejte dohromady a nasypte je do jednoho tabulkového turbokompresoru.

Rychlé vykreslování tabulky

V prohlížečích, které podporují CSS2[7], jako jsou IE6 a NS6, mohou být tabulky významně urychleny s použitím nové vlastnosti *table-layout*. Tato vlastnost umožňuje designérům rozhodnout, jaký algoritmus prohlížeč použije pro zalomení buněk, řádků a sloupců.

Pevný layout tabulky

Prohlížeč využívající nejrychlejší fixovaný algoritmus *table-layout* nenastavuje vodorovný rozměr tabulky podle obsahu buněk, ale podle šířky tabulky, sloupců, okrajů a mezer. Tento algoritmus je rychlejší, protože vodorovný zlom tabulky není závislý na obsahu každé buňky, ale na definovaných nebo výchozích šířkách. Takže prohlížeč zobrazuje tabulku již při prvním průchodu a nemusí tak čekat na načtení zbývajících části tabulky. To je užitečné zejména u delších tabulek.

Jak funguje fixní návrh tabulky? V každém sloupci určuje první buňka, jejíž šířka není nastavena na automatickou hodnotu, šířku daného sloupce. Následující buňky v daném sloupci jsou pak nastaveny na stejnou šířku, bez ohledu na jejich obsah. Při použití fixního algoritmu pro zobrazení tabulky musí být první buňka v každém sloupci nejširší.

Vlastnost *table-layout* může nabývat tří hodnot:

- auto (implicitní hodnota)
- fixed
- inherit

Například:

```
table { table-layout: fixed }
col.sum { width: 10em }
```

Použití výchozí hodnoty *auto* může být neefektivní. Vyžaduje až dva průchody prohlížeče pro správné zjištění rozměrů tabulky, protože musí zpracovat celý obsah tabulky.

Pokud nastavíte vlastnost *table-layout* na *fixed*, nastavíte pevné šířky sloupců (a volitelně i jejich pevné výšky) pro celou tabulku. Šířky sloupců jsou odvozeny od šířek definovaných prvky *col* (neautomatické hodnoty) nebo pokud nejsou použity, podle první buňky v každém sloupci. Bez prvků *col* je prohlížeč schopen zalamovat tabulku ihned, jakmile dostane data prvního řádku. S prvky *col* již není tento krok potřebný. Buňky v následujících řádcích již neovlivňují šířky sloupců.

V prohlížečích, které podporují CSS2, dokáže takové nastavení podstatně zvýšit výkon zpracování a zobrazování tabulek. Microsoft prohlašuje, že použití vlastnosti *table-layout* nastavené na hodnotu *fixed* přináší až 100násobné urychlení v Internet Exploreru 6 a vyšším. Pevný algoritmus umožňuje postupné zobrazování tabulky, najednou po řádcích. Výpis 4.4 uvádí příklad.

Výpis 4.4 – Příklad tabulky s fixním layoutem

```
<style type="text/css">
<!--
        table#skel {table-layout:fixed;width:100%;}
        col#left {width:33%;}
        col#right {width:67%;}
        tbody#main {text-align:center;}
-->
</style>
</head><body>
<table id="skel" frame="border">
<colgroup>
        <col id="left">
        <col id="right">
</colgroup>
        <tr>
                <td colspan="2">
                        <p>horní navigační řádek (název webu a reklama)</p>
                </td>
        </tr>
        <tbody id="main">
        <tr>
                <td>
                        <p>levy navigační sloupec</p>
                </td>
                <td>
                        <p>oblast pro hlavni obsah stranky</p>
                </td>
        </tr>
```

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

Řízení přetečení

Pro jistotu je potřeba ověřit, zdali nejsou všechny ostatní buňky ve sloupci širší než první buňka. Pokud obsah přesahuje buňku, je zalomen, nebo pokud zalomení není možné, je ořezán. V CSS2 existuje vlastnost *overflow*, která umožňuje ovlivnit chování buněk v případě, že obsah přesahuje velikost jakéhokoli bloku, včetně prvku *td*. Zde je příklad použití:

```
<style type="text/css">
<!--
        table#fixed {table-layout:fixed;width:100%;overflow:hidden;}
-->
</style>
</head>
<body>
<table id="fixed">
```

Vlastnost *overflow* může nabývat čtyř hodnot:

- **visible** – obsah, který se nevejde do svého bloku, se zobrazí (výchozí hodnota)
- **scroll** – do bloku se vloží rolovací lišta (scrollbar) bez ohledu na přetečení
- **hidden** – obsah, který se nevejde do svého bloku, bude ořezán
- **auto** – obsah, který se nevejde do svého bloku, bude vizuálně ořezán a přidá se rolovací lišta pro zobrazení skrytého obsahu

Tabulka s fixním layoutem v praxi

Na WebReference.com jsem zkoušel použít algoritmus s vlastností *table-layout* nastavenou na hodnotu *fixed*. Dosáhl jsem smíšených výsledků. Výpis 4.5 uvádí část ze zdrojového kódu.

Výpis 4.5 – Tabulka s fixním layoutem v praxi

```
<style type="text/css">
...
TABLE#f{table-layout:fixed;width:100%}
COL#a{width:9px;}
COL#b{width:134px;}
COL#c{width:9px;}
COL#d{width:65%;}
COL#e{width:8px;}
COL#f{width:1px;}
COL#g{width:125px;}
```

```
</style>
...
<TABLE id=f WIDTH="100%" BORDER=0 CELLSPACING=0 CELLPADDING=0>
<colgroup><col id=a><col id=b><col id=c><col id=d><col id=e><col id=f><col
id=g></colgroup>
<TR VALIGN=TOP><TD WIDTH=9 BGCOLOR="#FFCC00"><IMG SRC="/t.gif" WIDTH=9 HEIGHT=1
ALT=""></TD>
... 134 px TD následují...
```

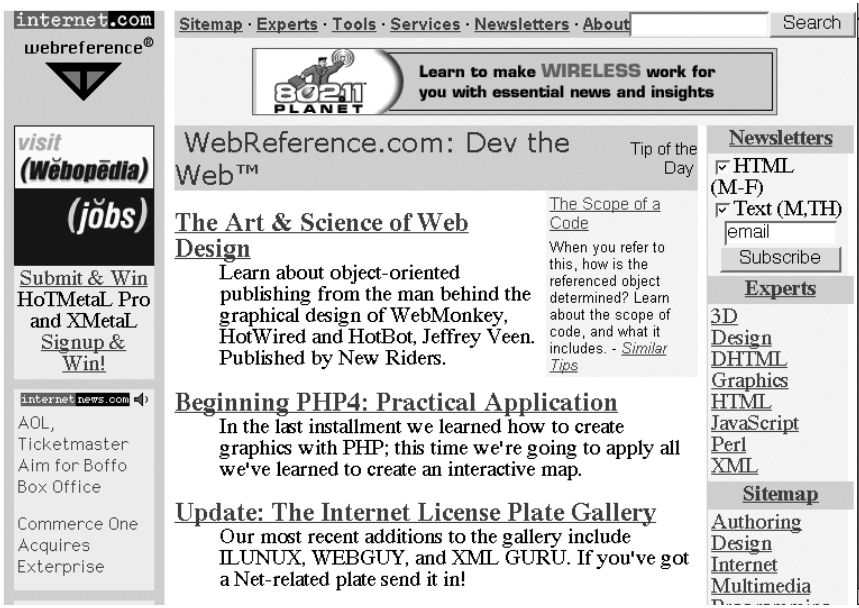
Poté, co jsem nastavil správně úvodní buňky, tabulka na obrazovce doslova lítala. Bohužel, narazil jsem na dva problémy. Vložené tabulky s reklamou bez zadaných rozměrů občas skočily dolů a v NS6.x texty přetékaly při uživatelském zvětšení písma na hodnotu "Větší" (larger). První problém se podařilo eliminovat přidáním *valign="top"*. Druhý byl o něco zapeklitější.

Uživatelské zvětšení písma v Netscape 6.x občas rozlije text mimo pevné sloupce, i když se při opětovném načtení stránky tento problém již neprojeví. IE5 pro Mac a IE6 pro Windows pracují se zvětšenými písmy lépe. Přetečení je možné korigovat použitím vlastnosti *overflow*, nicméně to nemá v NS6.x žádný efekt. Pokud použijete striktní DOCTYPE, vlastnost *overflow* bude aplikována na celý objekt HTML.

Obrázek 4.1 ukazuje, co se stane v Netscape 6.2, pokud použijete tabulku s fixním layoutem (*table { table-layout: fixed }*) a pak v prohlížeči zvětšíte velikost písma.

Obrázek 4.1 – Tabulka s fixním layoutem v Netscape 6.2.

Tento problém částečně vyřeší zvětšení původních rozměrů buněk. Nicméně, v Netscape 7 je tato chyba v přetékání a v algoritmu fixované tabulky při použití zvětšeného písma již opravena. Viz obrázek 4.2.



Obrázek 4.2 – Tabulka s fixním layoutem v Mozille 1.0 (Netscape 7).

Pokud používáte v tabulkách obsah s pevnou šířkou, například obrázky, použijte algoritmus *fixed*. Pokud má mít obsah buněk proměnlivou šířku, otestujte tabulku i při zvětšeném písmu.

Osobně se mi osvědčila specifikace šířky prostřednictvím atributu *col*. Zde je příklad z úvodní stránky WebReference.com:

```
<table id=f width="100%" border=0 cellspacing=0 cellpadding=0>
<colgroup><col width=9><col width=134><col width=9><col width="65%"><col
width=8><col width=1><col width=126>
```

Nebo lépe s využitím stylů:

```
table#f{table-layout:fixed;width:100%}
table#f{overflow:auto;}
col#a{width:9px;}
col#b{width:134px;}
col#c{width:*;}
col#d{width:8px;}
col#e{width:1px;}
col#f{width:126px;}
```

Výhodným kompromisem je podmíněné vkládání fixních a "přetékajících" (overflow) stylů pro prohlížeče, které podporují CSS2 – jako třeba IE6 a vyšší nebo Netscape 7 a vyšší, který je odvozen z Mozilly 1.0. V delších tabulkách používejte *colgroup* a *col*. Všude, kde je to možné, použijte *table-layout* a hodnotu *overflow*.

Také nastavení výšky sloupce může vést ke zvýšení rychlosti vykreslování, protože syntaktický analyzátor (parser) prohlížeče může začít zobrazovat řádek bez předchozího zkoumání obsahu všech buněk v řádku, aby zjistil, jaká je výška řádku.

Zjednodušujte

Mám pocit, že na všech školeních o sebezdokonalení osobnosti řečníci nabízejí samé zjednodušování. Zjednodušte svůj život; odstraňte vše, co není podstatné. Snižte hladinu vašeho stresu. Berte Echinaceu. Dýchejte! Stejné rady se hodí i na tabulky (možná s výjimkou dýchání a echinaceových kapek).

Prohlížeče obvykle zpracovávají tabulky v několika fázích. Nejprve se podívají do buněk, aby se podle obsahu mohla stanovit velikost. Pak dynamicky nastavují velikost buněk tak dlouho, dokud není spočítána celá tabulka. Odstraňte z tabulek vše, co není podstatné. Odstraňte všechny ikonky, které jste přidali někdy vloni. Zahoďte tyto reklamní a členské odkazy, které vám stejně nepřinášejí žádné peníze. Zbavte se pomalu stahujících javových tickerů se zprávami. Není to lepší?

Nechejte pouze nové a aktuální věci. Dokonce budu souhlasit s ponecháním vašeho loga určeného k propagaci vaší značky. Nyní budou vaše tabulky opravdu sprintovat. Zpět vložte pouze to, co chtějí vaši uživatelé nebo ještě lépe – nechejte je rozhodnout. Odstranění takových přebytečností je v tomto případě jedním z nejefektivnějších způsobů urychlování tabulek a webových stránek.

Vnořování tabulek

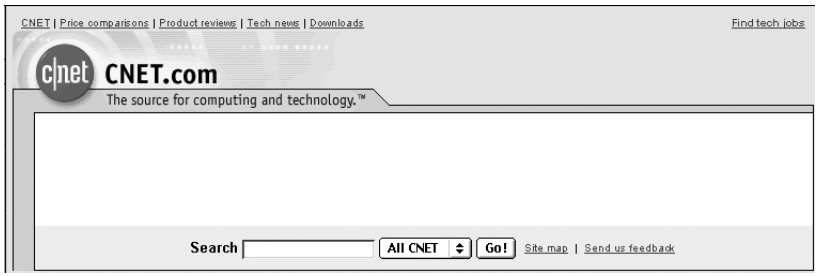
Čím hlouběji vnoříte tabulky do jiných tabulek, tím pomaleji se budou zobrazovat. Po jejich kompletním stažení musí prohlížeč provést spoustu výpočtů, aby správně zjistil jejich velikost. Mnohokrát vnořované tabulky zaměstnávají procesory uživatelů, zejména, pokud patří mezi ty pomalejší. Protože prohlížeč musí jako první spočítat nejhlouběji vnořenou tabulku, a pak se postupně vracet zpět k rodičovským (nadřazeným) tabulkám, je nejlepší vytvářet takový kód, který maximálně redukuje počet vnořených tabulek a úrovně vnoření.

Rozdělení tabulek

Na mnoha webech se používá jedna jediná tabulka pro vložení celého obsahu stránky. V naprosté většině případů to znamená, že prohlížeč tuto tabulku zobrazí až po kompletním stažení jejího rozsáhlého obsahu a po dokončení výpočtu rozměrů. To dost podstatně zpomaluje její zobrazení. Řešení je jednoduché – rozdělit takové tabulky na více samostatných tabulek – jako řezy ovocného koláče.

Vnímaná rychlost

Na horní část stránky použijte jednoduchou a rychle se stahující tabulku. Do této první tabulky vložte užitečný obsah, jako například vyhledávací box nebo navigační pruh, takže se uživatelé mohou rychle zorientovat. Prohlížeče zobrazují samostatné tabulky postupně, takže nejprve nabídněte něco, na co se uživatelé mohou dívat, zatímco se stahuje zbytek stránky. Takové stránky působí na pohled rychlejším dojmem, i když se mohou celkově stahovat pomaleji. Na obrázku 4.3 je příklad stránky CNET.com.



Obrázek 4.3 – Horní navigační pruh na CNET.com

Bez přehánění se dá říci, že první tabulka doslova vystřelí na stránku a dává uživateli něco, s čím může okamžitě pracovat. V kapitole 1 bylo řečeno, že velmi důležitá je zpětná vazba. Pokud se okamžitě objeví nějaký užitečný obsah, uživatelé pravděpodobněji kliknou na vaše další odkazy. Na obrázku 4.4 je příklad z úvodní stránky WebReference.com.

Obrázek 4.4 – Navigační pruh na WebReference.com

Na druhé straně, pokud se objeví něco, co není důležité, bude to uživatele nudit. Přednostním zobrazením loga a reklamy uživatel nedostává žádný užitečný obsah a nemůže se hned na stránce orientovat. Obrázek 4.5 ukazuje příklad:

Obrázek 4.5 – Onlinenewspapers.com.

Návrh stránky se třemi sekcemi

Podobná strategie zahrnuje změnu designu tabulek ke zvýšení hodnoty stránky pro vyhledávací servery a zároveň viditelnému zrychlení zobrazování. Typická stránka se třemi sekcemi obsahuje nahoře označení firmy (logo), eventuálně reklamu, v levém sloupci navigaci a v pravém sloupci hlavní obsah stránky. Výpis 4.6 uvádí příklad.