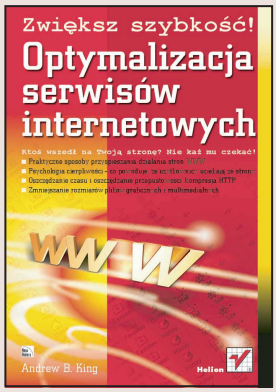
Zwiększ szybkość! Optymalizacja serwisów internetowych



Autor: Andrew B. King
Tłumaczenie: Marek Suczyk (rozdz. 1 – 11),
Jacek Smycz (rozdz. 12 – 19)
ISBN: 83-7361-134-7
Tytuł oryginału: Speed Up Your Site – Web Site Optimization
Format: B5, stron: 416
Przykłady na ftp: 80 kB

Cierpliwość użytkownika jest jak bomba zegarowa. Każde wejście na Twoją stronę WWW uruchamia odliczanie. Masz zaledwie kilka sekund, by dostarczyć odwiedzającemu treści, których poszukuje. Jeśli nie zdążysz, możesz się pożegnać ze swoim gościem (i z ewentualnymi zyskami, które osiągnąłbyś dzięki jego wizycie). Nie możesz liczyć na szybkie łącza. W dalszym ciągu wielu gości Twojej witryny używa sieci telefonicznej i zwykłych modemów. Albo zdążysz, nim skończy się ich cierpliwość, albo przegrasz. Dlatego powinieneś sięgnąć po tę książkę. Dowiesz się z niej, jak skrócić o połowę czas wczytywania strony. Zmniejszysz rozmiary plików HTML, XHTML, CSS, ilustracji i skryptów JavaScript oszczędzając dodatkowo na kosztach łączy. Przykłady wzięte z praktyki przedstawiają techniki przynoszące rzeczywiste rezultaty. Po przeczytaniu tej książki będziesz już wiedział, jak tworzyć strony pojawiające się na ekranie w mgnieniu oka.

Nauczysz się:

- Przyspieszać ładowanie się stron WWW
- Angażować użytkowników w proces przeglądania strony
- Analizować efekty psychologiczne wywoływane wolnym ładowaniem się stron
- Zmniejszać rozmiary i stopień skomplikowania plików HTML
- Stosować skróty w CSS
- Przyspieszać i odchudzać JavaScript
- Zmniejszać rozmiary plików graficznych i multimedialnych
- Oszczędzać przepustowość łączy (aż do 60%!) stosując kompresję HTTP

Andrew B. King (Andy) jest założycielem WebReference.com oraz JavaScript.com (obie witryny zostały nagrodzone w klasyfikacji stron przeznaczonych dla programistów). WebReference.com – utworzona w 1995 roku, a następnie w 1997 r. zaadoptowana przez Macklermedia (obecnie Jupitermedia) – urosła do rangi jednej z najpopularniejszych stron internetowych dla programistów.

Ktoś wszedł na Twoją stronę? Nie każ mu czekać!

- Praktyczne sposoby przyspieszania stron WWW
- Psychologia cierpliwości – co powoduję, że użytkownicy uciekają ze strony
- Oszczędzanie czasu i oszczędzanie przepustowości: kompresja HTTP
- Zmniejszanie rozmiarów plików graficznych i multimedialnych.

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Rozdział 8.

Zaawansowana optymalizacja kodu CSS

Oprócz omówionych korzyści płynących z buforowania plików CSS i zmniejszenia obciążenia łącza, CSS pozwala na zastąpienie elementów strony, które pochłaniają znaczne zasoby i wydłużają czas ładowania (np. grafika nakładkowa), na „lekkie" elementy, nieobciążające strony. W tym rozdziale zamieszczone zostały informacje omawiające cztery główne sposoby wykorzystania arkuszy CSS do przyspieszenia stron internetowych. Zagadnienia te dotyczą:

- mniejszych rozmiarów arkuszy stylów,
- elementów zastępczych,
- szybszych tabel,
- kontroli układu i rozmieszczenia elementów.

Zasady optymalizacji arkuszy CSS

Na podstawie metod, których zdążyłeś się nauczyć w rozdziale 7., będziesz mógł zobaczyć, w jaki sposób przeprowadzana jest optymalizacja starego arkusza stylów strony *WebReference.com* (listing 8.1).

Listing 8.1. *Oryginalny arkusz stylów WebReference.com*

```
<style type="text/css">
<!--
form.tb{display:inline;}
.h{text-decoration:none;font-size:9pt;font-family:geneva,arial,sans-serif;}
.c{font-size:80%;font-family:arial,geneva,sans-serif;}
.d{font-size:70%;font-family:arial,geneva,sans-serif;}
dt{font-weight:bold;font-size:120%;margin-top:.8em;}
```

```
.w{font-size:125%;font-family:verdana,sans-serif;color:#660099;}
.NSlyr{width:119;position:absolute;visibility:hidden;}
a:hover{background-color:#ffdd33;}
-->
</style>
```

W powyższym arkuszu można zauważyć wiele technik omawianych we wcześniejszych rozdziałach książki. Używane są skróty, jednoznakowe nazwy klas (jak `.h` i `.c`), proste selektory typów (`dt`), pseudoklasa `:hover`. Arkusz nie jest połączony ze stroną, lecz jest w niej osadzony. Zawsze jednak można coś zrobić lepiej. Wykorzystajmy skrócone właściwości `font` i `background`, jak również skorzystajmy ze skróconego zapisu szesnastkowego:

```
<style type="text/css">
<!--
form.tb{display:inline}
.h{text-decoration:none;font:9pt geneva,arial,sans-serif}
.c{font:80% arial,geneva,sans-serif}
.d{font:70% arial,geneva,sans-serif}
dt{font:bold 120% serif;margin-top:.8em}
.w{font:125% verdana,sans-serif;color:#609}
.NSlyr{width:119;position:absolute;visibility:hidden}
a:hover{background:#fd3;}
-->
</style>
```

Dzięki wykorzystaniu skróconych właściwości oszczędzamy 99 bajtów (ilość znaków zmniejsza się z 449 do 350). Ponieważ skrócona właściwość `font` wymaga wartości `font-family`, może pojawić się problem z listą definicji, jeżeli użytkownik wybierze czcionkę `sans-serif`. Elementy `dt` będą wówczas pisane czcionką szeryfową, a elementy `dd` bezszeryfową. Aby temu zaradzić, można ustawić czcionkę dla elementu `body` na `serif`:

```
<style type="text/css>
<!--
body{font:1em serif}
form.tb{display:inline}
.h{text-decoration:none;font:9pt geneva,arial,sans-serif}
.c{font:80% arial,geneva,sans-serif}
.d{font:70% arial,geneva,sans-serif}
dt{font:bold 120% serif;margin-top:.8em}
.w{font:125% verdana,sans-serif;color:#609}
.NSlyr{width:119;position:absolute;visibility:hidden}
a:hover{background:#fd3;}
-->
</style>
```

Lecz jak się okazuje, dłuższy zapis właściwości `dt` i pozostawienie `font-family` jest w rzeczywistości krótsze o 4 znaki (360 zamiast 364 znaków):

```
<style type="text/css">
<!--
form.tb{display:inline}
.h{text-decoration:none;font:9pt geneva,arial,sans-serif}
.c{font:80% arial,geneva,sans-serif}
```

```
.d{font:70% arial,geneva,sans-serif}
dt{font-weight:bold;font-size:120%;margin-top:.8em}
.w{font:125% verdana,sans-serif;color:#609}
.NSlyr{width:119;position:absolute;visibility:hidden}
a:hover{background:#fd3;}
-->
</style>
```

W tym miejscu pierwotny arkusz stylów został zmniejszony o 20% (z 449 znaków do 360). W dalszej części optymalizacji można pogrupować rodziny czcionki na trzy style i osobno określić ich rozmiary:

```
<style type="text/css">
<!--
form.tb{display:inline}
.h,.c,.d{font-family:arial,geneva,sans-serif}
.h{text-decoration:none;font-size:9pt}
.c{font-size:80%}
.d{font-size:70%}
dt{font-weight:bold;font-size:120%;margin-top:.8em}
.w{font:125% verdana,sans-serif;color:#609}
.NSlyr{width:119;position:absolute;visibility:hidden}
a:hover{background:#fd3;}
-->
</style>
```

Ilość znaków zmniejszyła się do 348. Można uzyskać jeszcze lepszy rezultat używając ogólnych rodzin czcionki, mniejszych rozmiarów oraz skróconej właściwości `font` (listing 8.2).

Listing 8.2. *Zoptymalizowany arkusz stylów WebReference.com*

```
<style type="text/css">
<!--
form.tb{display:inline}
.h,.c,.d{font:80% sans-serif}
.h{text-decoration:none}
dt{font-weight:bold;font-size:120%;margin-top:.8em}
.w{font:125% sans-serif;color:#609}
.NSlyr{width:119;position:absolute;visibility:hidden}
a:hover{background:#fd3}
-->
</style>
```

Wielkość kodu została zmniejszona do 276 bajtów, czyli o 38 procent względem pierwotnego rozmiaru. Czy zwróciłeś uwagę na połączenie metody grupowania ze skróconą właściwością `font`? Zauważ, że powyższy kod można jeszcze bardziej zmniejszyć usuwając z niego wszystkie znaki końcowe i niepotrzebne spacje (pamiętaj, że linia nie powinna być dłuższa niż 255 znaków). Dzięki kaskadowaniu, dziedziczeniu, grupowaniu i skracaniu właściwości można zmniejszyć rozmiar arkusza CSS o ponad 50 procent.

Elementy zastępcze

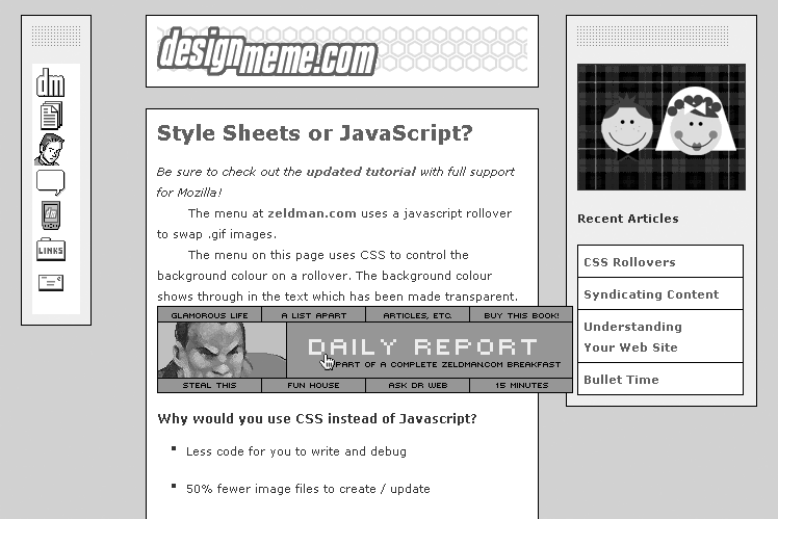
Coraz więcej przeglądarek obsługuje pseudoklasę `hover` arkusza CSS2. Dzięki temu można uzyskać efekty nakładkowe, które pobierają znacznie mniej zasobów niż JavaScript. Można wyróżnić dwa podstawowe sposoby optymalizacji elementów nakładkowych:

- za pomocą grafiki nakładkowej CSS2 — przeźroczyste GIF-y oraz kolorowe tła mogą o połowę zmniejszyć ilość potrzebnych obrazków,
- za pomocą samych nakładek CSS2 — obrazki zostają całkowicie zastąpione tekstem.

Bez względu na to, z którego sposobu skorzystasz, zawsze rezultatem będzie zmniejszenie ilości żądań HTTP i wielkości kodu.

Grafika nakładkowa CSS2

Grafika nakładkowa CSS2 polega na używaniu przeźroczystych GIF-ów oraz pseudoklasy `hover` do zmiany tła za obrazkiem nakładkowym. Efekt jest taki sam, jak podczas korzystania z dodatkowych obrazków i kodu JavaScript. Stuart Robertson z Designmeme.com przedstawił sposób ulepszenia paska narzędzi na stronie *Zeldman.com* (rysunek 8.1).



Rysunek 8.1. *Grafika nakładkowa CSS2 (http://designmeme.com/zeldman)*

Wszystko, co jest potrzebne, to zaledwie kilka obrazków z przeźroczystymi wycinkami oraz kilka wierszy kodu CSS:

```
.zeldman a { display:block; width:100px; background-color: #000000;}
.zeldman a:hover {background-color: #CCFF00}
```

Uproszczony kod HTML wygląda następująco:

```
<div class="zeldman">
<table border="0" cellpadding="0" cellspacing="0">
      <tr>
            <td bgcolor="#000000"><a href=http://www.zeldman.com/glamorous/>
                  <img name="glam" src="glam.gif" width="100" height="16" border=
                  "0" alt="MY GLAMOROUS LIFE: Tragicomic fodder from the life of
                  Zeldman."/></a></td>
...
```

Ta pomysłowa metoda zmienia kolor tła łącza za obrazkiem nakładkowym. Ponieważ tekst na obrazku jest przeźroczysty, przebija przez niego kolor tła. Oczywiście powyższy kod można lepiej zoptymalizować używając skróconej właściwości `background` i krótkich szesnastkowych zapisów kolorów. Teraz jednak chodzi o przedstawienie samego pomysłu. Opisana metoda znakomicie działa w starszych przeglądarkach i nie wymaga korzystania z kodu JavaScript.

Proste nakładki CSS

Możesz zrobić kolejny krok i wymienić wszystkie obrazki na CSS2. Proste nakładki CSS2 używają stylów zarówno dla pierwszoplanowych elementów menu, jak i dla samego tła menu. Autorzy stron znaleźli pomysłowy sposób na przekształcenie łączy w nakładki. Niektórzy tworzą przyciski trójwymiarowe, inni przekształcają łącza w elementy blokowe, a jeszcze inni zmieniają listy w wewnętrzne elementy wierszy. Bez względu na to, który sposób wybierzesz, wszystkie metody bazują na pseudoklasie `hover`, umożliwiającej utworzenie prostej nakładki tekstowej.

Proste nakładki tekstowe

Najprostszym sposobem dodania efektu nakładkowego do łącza jest użycie pseudoklasy `hover`. Łącza są tworzone w normalny sposób, lecz dodawany jest do nich kolor tła:

```
a:hover{background:#fd3}
```

Z technicznego punktu widzenia powyższa metoda może mieć wpływ na wszystkie zakotwiczone obiekty (nie tylko na łącza) i wszędzie wyświetlać efekt `hover`. Dokładniejsza definicja stylu `hover` dla łączy jest następująca:

```
a:link:hover{background:#fd3}
a:visited:hover{background:#fd3}
```

Dodatkowe informacje

Więcej informacji o grafice nakładkowej CSS można znaleźć na następujących stronach:

- *http://www.alistapart.com/stories/rollovers/*
- *http://www.designmeme.com/zeldman/*

W momencie, gdy użytkownik wskaże łącze, zgodnie z powyższą zasadą kolor tła łącza zmieni się na żółty. Aby mieć pewność, że zmiana koloru tła nie zmniejszy czytelności wyświetlanego tekstu, należy również określić kolor pierwszoplanowy:

```
a:link:hover{background:#fd3;color:#00f}
a:visited:hover{background:#fd3;color:#00f }
```

Zebranie wszystkich pseudoklas łączy może zostać zapisane w następujący sposób:

```
a:link             { color: #00f } /* nieodwiedzane łącza - niebieski */
a:visited          { color: #609 } /* odwiedzane łącza - szkarłatny */
a:link:hover       { color: #fd3 } /* wskazane przez użytkownika - żółte */
a:visited:hover    { color: #fd3 } /* wskazane przez użytkownika - żółte */
a:link:active      { color: red } /* aktywne łącza */
a:visited:active   { color: red } /* aktywne łącza */
```

Możesz uzyskać odwrotny efekt poprzez zamianę koloru pierwszoplanowego z kolorem tła.

```
a:link             { color: #00f; background:#fc0 }
a:visited          { color: #609 }
a:link:hover       { color: #fc0; background:#00f }
a:visited:hover    { color: #fc0; background:#00f }
a:link:active      { color: red } /* aktywne łącza */
a:visited:active   { color: red }
```

Pionowe menu CSS2

Na stronie *WebReference.com* utworzone zostały przykładowe pionowe nakładki CSS2, które nie są prawidłowo wyświetlane w Netscape 4 (*http://www.webreference.com/new/rollovers*). W przeglądarce Netscape 4 wyglądają niemal identycznie, lecz nie mają efektu nakładkowego. W przeglądarkach obsługujących pseudoklasę `hover` (Opera 3.5, IE4+, Netscape 6+/Mozilla) działają poprawnie.

Po dokładnych testach znaleźliśmy rozwiązanie, które działało prawidłowo we wszystkich przeglądarkach biorących udział w testach. Użyty został znacznik łącza dla arkuszy stylów. Wyniki zostały przedstawione na rysunku 8.2. Listing 8.3 prezentuje kod CSS.

Łączenie pseudoklas

Internet Explorer dla Windows może mieć pewne problemy z łańcuchami pseudoklas. Wygląda na to, że wersje IE4, IE5 i IE6 mogą ignorować wszystkie, z wyjątkiem ostatniej pseudoklasy w łańcuchu. Dzieje się to tak długo, dopóki nie pojawi się identyfikator elementu. Na szczęście w tym przypadku nie stanowi to problemu, gdyż `a:link:hover` i `a:visted:hover` mają określony ten sam styl (Internet Explorer efektywnie skraca oba zapisy do `a:hover`). To samo dotyczy `:active`. Internet Explorer może zachowywać się dziwnie, gdy zechcesz nadać różne style połączonym pseudoklasom, albo gdy nadasz styl tylko części z nich. Większość projektantów nie ma problemu z takim zachowaniem, gdyż używa prostszego selektora `a:hover` albo selektora kontekstowego. Więcej informacji znajdziesz pod adresem *http://www.meyerweb.com/eric/css/tests/css2/sec05-10.htm*.

Rysunek 8.2.
Proste pionowe nakładki CSS2

Listing 8.3. *Pionowe nakładki CSS2 (kod CSS)*

```
body {background:#fff;}

h4 {margin:0;padding:0.3em;text-align:center;}

div.menu {
      width:125px;
      background:#fff;
      padding:0;
      margin:1em;
      border:1px solid #000;
}

div.menu a {
      display:block;
      margin:0;
      width:100%;
      padding:0.3em;
      font-weight:bold;
      border-top:1px solid #000;
      color:#00f;
      text-decoration:none;
}

html>body div.menu a {width:auto;}

div.menu a:hover {background:#000;color:#fff;}
```

Aby uzyskać efekt nakładkowy na całej szerokości menu, wykorzystana została sztuczka zapożyczona z *www.tantek.com*. Szerokość łącza jest ustawiana na `auto` dla przeglądarek innych niż IE6:

```
html>body div.menu a {width:auto;}
```

Autorzy, którzy nie chcą korzystać z tej sztuczki, mogą napisać `width:auto;` dla łączy blokowych. Internet Explorer nałoży nakładkę tylko na łącza. Wszystkie pozostałe przeglądarki zgodne z CSS2 będą poprawnie obsługiwać cały pojemnik, a łącze będzie aktywne.

Spacje na listingu 8.4 zostały dodane z myślą o zachowaniu łączy w starszych przeglądarkach. Na zewnątrz łączy znaczniki `<br>` mogą zostawiać pionową lukę. Aby uniknąć wyświetlania luk, należy przenieść je na zewnątrz łączy i nie stosować stylu dla przeglądarek niezgodnych z CSS:

```
div.menu br {display:none;}
```

Listing 8.4. *Pionowe nakładki CSS (kod HTML)*

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Proste nakładki CSS2</title>
<link rel="stylesheet" href="listing8.3.css" type="text/css">

</head>
<body>

<div class="menu">
<h4>Menu Title</h4>
<a href="/products/">Products <br></a>
<a href="/services/">Services <br></a>
<a href="/about/">About <br></a>
<a href="/contact/">Contact <br></a>
</div>

</body>
</html>
```

Aby najwyższy element menu był również aktywny, można zmienić nagłówek w styl łącza, analogicznie do pozostałych elementów:

```
<div class="menu">
<a href="/home/">Home <br></a>
<a href="/products/">Products <br></a>
```

Pojawia się jednak inny problem — górna krawędź menu jest rysowana podwójną linią, gdyż zarówno zewnętrzny `div`, jak i łącze `home` mają krawędź górną (rysunek 8.3).

Rysunek 8.3. *Proste pionowe nakładki CSS2 — zwróć uwagę na podwójną krawędź górną*

Aby pozbyć się dodatkowej linii, wystarczy usunąć ją z górnej krawędzi `div`, tak jak zostało to przedstawione poniżej (spójrz na rezultat na rysunku 8.4):

```
div.menu {
      width:125px;
      background:#fff;
      padding:0;
      margin:1em;
      border:1px solid #000;
      border-top:0px; /* usuń podwójną linię */
}
```

Rysunek 8.4. *Proste pionowe nakładki CSS2 — z pojedynczą górną krawędzią*

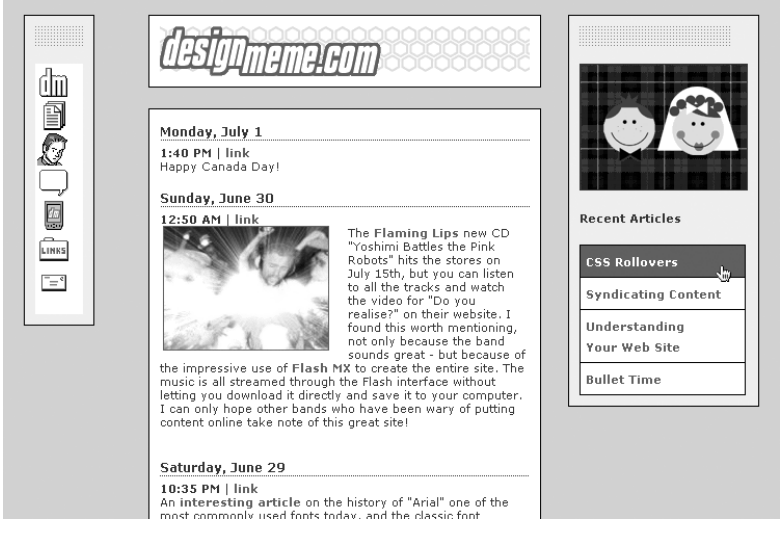
W przypadku korzystania z poleceń CSS2, które mogą być przyczyną potencjalnych problemów w Netscape 4, można opcjonalnie zastosować dyrektywy `@import`, aby czytać z zewnętrznego arkusza stylów. Powyższe przykłady zostały przedstawione dla czarno-białych kolorów. Generalnie nie zalecam używania czarnych łączy. Nawet w menu użytkownicy spodziewają się domyślnego, niebieskiego koloru łączy. Łącza powinny odróżniać się od zwykłego tekstu.

Prawdziwy przykład nakładek CSS2

Projektanci zamieniają stare nakładki bazujące na grafice i kodzie JavaScript na nowe, „lekkie" menu CSS2. Przyjrzyjmy się prawdziwym przykładom korzystającym z nakładek CSS2. Zacznijmy od *Designmeme.com*.

Designmeme.com

Stuart Robinson, webmaster na Uniwersytecie Guelph w Kanadzie, na swojej stronie internetowej używa dwóch typów nakładek CSS2, które zostały przedstawione na rysunku 8.5. Z lewej strony widoczne jest menu utworzone za pomocą przeźroczystych GIF-ów i pseudoklasy `hover` (bez korzystania z nakładek graficznych JavaScript). Pionowy pasek menu, widoczny po prawej stronie, korzysta natomiast z prostych nakładek CSS2, bez żadnej dodatkowej grafiki.

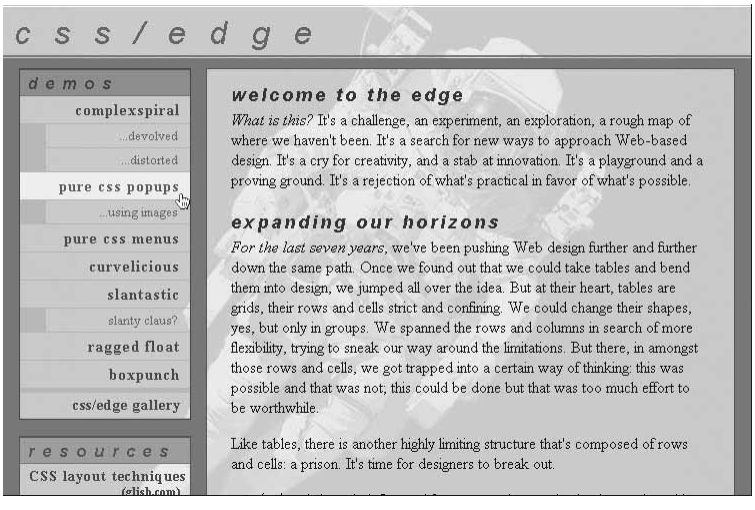


Rysunek 8.5. *Designmeme.com*

Stuart Robinson jest pionierem w dziedzinie nakładek CSS, których używa od maja 2001 r.

Strona CSS Edge Erica Meyera

Eric Meyer jest autorem książek: „Cascading Style Sheets: The Definitive Guide" (O'Reilly, 2000 r.) („CSS. Kaskadowe arkusze stylów. Przewodnik encyklopedyczny", Helion, 2001 r.) oraz „Eric Meyer on CSS" (New Riders, 2002 r.). W swoich książkach omawia wiele technik używania arkuszy CSS. Na jego stronie *CSS Edge* można znaleźć wiele technik opartych na arkuszach stylów CSS, które mogą poprawnie pracować w nowszych przeglądarkach (rysunek 8.6).



Rysunek 8.6. *Strona CSS Edge Erica Meyera (http://www.meyerweb.com/eric/css/edge/)*

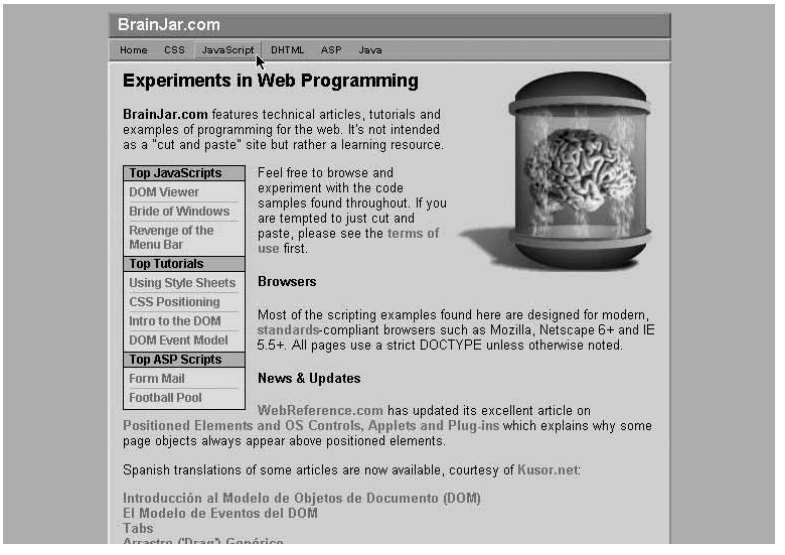
Poziome nakładki CSS2

Używając podobnych metod i bez stosowania sztuczki `display:block`, można utworzyć poziome nakładki CSS2. Mike Hall na stronie *Brainjar.com* przedstawia bardzo dobry przykład poziomego paska menu bazującego na nakładkach CSS2 (rysunek 8.7). Poprzez wykorzystanie offsetu i efektu jasnych krawędzi (ang. *light-sourced*) przyciski menu zyskały trójwymiarowy wygląd.

Mike Hall poszedł krok dalej, dodając hierarchiczne rozwijane menu korzystające z `div`-ów i odrobiny JavaScript. Menu jest poprawnie wyświetlane w Netscape 6 i Internet Explorerze 5.5, lecz nie sprawdza się w IE5 Mac i Operze[1] (rysunek 8.8).

Dzięki Kwon Ekstrom i za pozwoleniem Mike'a Halla możemy zajrzeć do środka trójwymiarowego menu, aby wyciągnąć z niego to, co najistotniejsze. Zacznijmy od HTML:

[1] Opera 7.x radzi sobie już z opisaną techniką dość dobrze — *przyp. red.*



Rysunek 8.7. *Strona BrainJar Mike'a Halla (http://www.brainjar.com/)*

Rysunek 8.8. *Strona BrainJar. Hierarchiczne menu oparte na modelu DOM*

```
<div class="menubar" width="100%">
<a class="button active" href="/">Home</a>
<a class="button" href="/products/">Products</a>
<a class="button" href="/services/">Services</a>
<a class="button" href="/contact/">Contact</a>
</div>
...
```

Jedyną niestandardową rzeczą w powyższym zapisie jest klasa `active`. Zoptymalizowany arkusz stylów na listingu 8.5 definiuje pasek nawigacyjny i sprawia, że aktywny przycisk wygląda tak, jakby był „wciśnięty”.

Listing 8.5. *Trójwymiarowe poziome menu CSS2*

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

div.menubar, div.menubar a.button {
      font: bold .9em arial,helvetica,sans-serif;
      text-decoration: none;
      color: blue;
}

div.menubar {
      background: #fd0;
      padding: 4px 2px;
      border: 2px solid;
      border-color: #ff9 #777 #777 #ff9;
      text-align: left; /* podczas środkowania dla ie */
}

div.menubar a.button {
      background: transparent;
      border: 1px solid #fd0;
      cursor: default;
      left: 0px;
      top: 0px;
      margin: 1px;
      padding: 1px 4px;
      position: relative;
      z-index: 100;
}

div.menubar a.button:hover {
      background: transparent;
      border-color: #ff9 #993 #993 #ff9;
      color: blue;
}

div.menubar a.active,
div.menuBar a.active:hover {
      background: #777;
      border-color: #333 #ff9 #ff9 #333;
      color: #fff;
      left: 1px;
      top: 1px;
}
```

Zasada `a.button` i odpowiadające jej style `hover` są kluczem do sukcesu. Przycisk jest definiowany tym samym kolorem tła (przeźroczysty), co otaczający go `div`, dzięki czemu „wtapia" się on w pasek nawigacyjny. Ustawiając offset i cień, można zasymulować wygląd wciśniętego przycisku (rysunek 8.9). Styl `a.button:hover` zmienia kolor wszystkich czterech krawędzi, nadając trójwymiarowy wygląd. Aktywny styl `hover` przesuwa przycisk o 1 piksel w dół i w prawo, zamieniając kolory krawędzi i przyciemniając tło. Więcej informacji o pasku menu znajdziesz na stronie *BrainJar.com* w części „Revenge of the Menu Bar".

Rysunek 8.9. *Pasek menu CSS2*

Dynamiczny pasek menu CSS2

Można połączyć ideę interaktywnego trójwymiarowego przycisku z warunkowym SSI i utworzyć dynamiczny pasek menu oparty na CSS2. Zacznijmy od prezentacji kodu HTML paska menu i odpowiadającemu arkuszowi CSS (listing 8.6).

Listing 8.6. *Szablon HTML paska menu CSS2*

```
<html>
<head><title>Demo paska menu CSS</title>
<style type="text/css">
<!--
@import "/css/menubar.css";
-->
</style>
</head>
<body>
<!--#include virtual="/css/menubar2.html" -->
</body>
</html>
```

Powyższy kod pozwala na umieszczenie na stronie paska menu o szerokości strony. Konieczna jest aktualizacja tylko dwóch plików. Zwróć uwagę, że dołączony zostaje plik HTML, a nie plik tekstowy. Poprzez ustawienie flagi w pliku konfiguracyjnym serwera (patrz rozdział 17.) można dołączyć warunkowy SSI w każdym dołączonym pliku HTML (listing 8.7).

Listing 8.7. *Warunkowe SSI paska menu*

```
<!--#if expr="(${DOCUMENT_URI} = /^\/products\/.*/)" -->
<div class="menubar" width="100%">
<a class="button" href="/">Home</a>
<a class="button active" href="/products/">Products</a>
<a class="button" href="/services/">Services</a>
<a class="button" href="/contact/">Contact</a>
</div>

<!--#elif expr="(${DOCUMENT_URI} = /^\/services\/.*/)" -->
<div class="menubar" width="100%">
<a class="button" href="/">Home</a>
<a class="button" href="/products/">Products</a>
<a class="button active" href="/services/">Services</a>
<a class="button" href="/contact/">Contact</a>
</div>

<!--#elif expr="(${DOCUMENT_URI} = /^\/contact\/.*/)" -->
<div class="menubar" width="100%">
<a class="button" href="/">Home</a>
<a class="button" href="/products/">Products</a>
<a class="button" href="/services/">Services</a>
<a class="button active" href="/contact/">Contact</a>
</div>
```

```
<!--#elif expr="((${DOCUMENT_URI} = /^\/$/) || (${DOCUMENT_URI} =
/^\/index\.html/))" -->
<div class="menubar" width="100%">
<a class="button active" href="/">Home</a>
<a class="button" href="/products/">Products</a>
<a class="button" href="/services/">Services</a>
<a class="button" href="/contact/">Contact</a>
</div>
<!--#else -->
<div class="menubar" width="100%">
<a class="button" href="/">Home</a>
<a class="button" href="/products/">Products</a>
<a class="button" href="/services/">Services</a>
<a class="button" href="/contact/">Contact</a>
</div>
<!--#endif -->
```

Nakładkowy pasek menu będzie zmieniał wygląd aktywnego przycisku. Przycisk będzie aktywny, jeżeli kursor będzie znajdował się w danym menu hierarchicznym, wyświetlonym za pomocą tego przycisku. W arkuszu stylów z listingu 8.5 wystarczy zmienić lokalizację klasy „`active`" w HTML. Przykładowo, przy wybraniu pozycji „`contact`", warunkowy SSI z listingu 8.7 sprawdza bieżący URL i znajduje odpowiednią instrukcję:

```
<!--#if expr="(${DOCUMENT_URI} = /^\/contact\/.*/)" -->
```

Wyrażenie odpowiada każdemu adresowi URL zaczynającemu się od `/contact/`, jak np. `/contact/staff.html` (rysunek 8.10).

Rysunek 8.10. *Dynamiczny pasek menu CSS2*

To samo przypisanie `class` można uzyskać za pomocą kodu JavaScript, lecz — jak widać na powyższym przykładzie — można się bez niego obejść.

Menu oparte na liście

Christopher Schmitt, założyciel *BabbleList.com*, jest najprawdopodobniej pierwszą osobą, która opublikowała sztuczkę polegającą na uzyskaniu poziomego nakładkowego menu przy użyciu `display:inline` na elementach listy. Na rysunku 8.11 przedstawiony został przykład.

Rysunek 8.11. *Strona CSSBook.com Christophera Schmitta. Menu oparte na liście (http://www.cssbook.com/cssnav/css2.html)*

Różnica pomiędzy tą i wcześniej opisaną metodą polega na tym, że w tym przypadku elementy menu są oddzielone od siebie wewnętrznymi elementami listy. Korzyść jest taka, że aktywny jest cały pojemnik, który w całości jest łączem. Margines wokół łącza tworzy przestrzeń otaczającą tekst. Nie ma potrzeby, aby używać łączy elementów blokowych. Poniżej przedstawiony został kod HTML:

```
<div id="nav"><p>Navigation:</p><ul><li><a href="/ankle">Ankle</a></li><li><a href=
"/boat">Boat</a></li><li><a href="cupcake">Cupcake</a></li><li><a href="/double"
>Double</a></li><li><a href="/eatery">Eatery</a></li></ul></div>
```

W prosty i jasny sposób `div "nav"` otacza nieuporządkowaną listę. Wystarczy nadać styl wyświetlania listy (`inline` zamiast domyślnego `list-item`):

```
#nav li {
      padding: 0;
      margin: 0;
      display: inline; /* zmienia li z blokowego na liniowy */
      font-size: 0.9em;
      font-family: Verdana, Arial, Helvetica, sans-serif;
}
```

Następnie należy dodać trochę marginesu wewnętrznego i efekt `hover`:

```
#nav li a {
      display: inline;
      padding: 7px;
      margin: 0;
      color: #333;
      background-color: #ccc;
      font-size: 0.9em;
      font-family: Verdana, Arial, Helvetica, sans-serif;
      text-decoration: none;
}

#nav li a:hover {
      color: #fff;
      background-color: #666;
      font-size: 0.9em;
      font-family: Verdana, Arial, Helvetica, sans-serif;
      text-decoration: none;
}
```

Zauważ, że powyższy kod można zoptymalizować za pomocą skróconych właściwości i usunięcia paru nadmiarowych informacji. Końcowy, zoptymalizowany kod arkusza stylów został przedstawiony na listingu 8.8.

Listing 8.8. *Zoptymalizowane menu oparte na elementach listy*

```
#nav {
      position:absolute;
      top: 12px;
      left: 12px;
      color: #fff;
      background: transparent;
      font: 0.9em verdana, arial, helvetica, sans-serif;
}
```

```
#nav ul, ul {
      margin: 0;
      padding: 0;
}

#nav li {
      padding: 0;
      margin: 0;
      display: inline;
      font: 0.9em verdana, arial, helvetica, sans-serif;
}

#nav li a {
      display: inline;
      padding: 7px;
      margin: 0;
      color: #333;
      background: #ccc;
      font: 0.9em verdana, arial, helvetica, sans-serif;
      text-decoration: none;
}

#nav li a:hover {
      color: #fff;
      background: #666;
}

#nav li a:active {
      color: #ccc;
      background: #333;
}
```

Tabele i CSS

W rozdziale 4. „Zaawansowane metody optymalizacji" dowiedziałeś się, w jaki sposób należy przyspieszać wyświetlanie tabel za pomocą właściwości CSS2 `table-layout:fixed` oraz elementów `colgroup` i `col`. Innym sposobem jest wykorzystanie arkuszy CSS do określenia koloru tła komórek tabeli. Dzięki temu nie trzeba w każdym miejscu pisać `bgcolor`. Zamiast:

```
<table bgcolor="#ffcc00">
<tr bgcolor="tensamkolor">
<td>...</td>
<td bgcolor"innykolor">...</td>
```

Dodatkowe informacje

Więcej informacji o menu opartym na wewnętrznych elementach listy można znaleźć na stronie Christophera Schmitta *CSSBook.com* albo w jego książce „Designing CSS Web Pages", New Riders, 2002 r. Polecam również stronę Dave'a Lindquista, omawiającą menu DHTML tworzone na bazie list (rozwijane i rozszerzalne poniżej 6 kB) — *http://www.gazingus.org/dhtml/?id=109*.

należy napisać:

```
.x{background:#fc0}.y{background:#ffed9a}.z{background:#fff3ac}

<table class="x">
<tr class="y">
<td>...</td>
<td class="z">...</td>
```

Wszystkie przeglądarki od wersji 4. obsługują powyższą metodę.

Kontrolowanie układu za pomocą arkuszy CSS

Rozmieszczenie elementów strony za pomocą arkuszy CSS może przekształcić stary tabelaryczny układ w elegancki układ bazujący na CSS. Wielu projektantów przekonało się, że zmiana sposobu rozmieszczenia elementów może zmniejszyć obciążenie łącza oraz koszty utrzymania. Pozbycie się tabel daje również większą elastyczność, pozwalając na wyświetlanie różnej zawartości z tymi samymi stylami. Przenosząc warstwę prezentacyjną do arkuszy stylów, struktura dokumentu staje się przejrzysta. Zmiana układu strony sprowadza się do zmiany jednego arkusza stylów. Przeniesienie zawartości na inną platformę (inną przeglądarkę lub wersję przeglądarki) albo inny typ mediów wymaga jedynie zastosowania innego arkusza stylów i nie zmusza do ponownego przepisywania kodu dokumentu.

Najpopularniejszym układem jest struktura dwu- lub trójkolumnowa (zachowanie zgodności z Netscape 4 jest tutaj pewnym wyzwaniem). Spójrzmy na przykładowy układ trójkolumnowy z listingu 8.9.

Listing 8.9. *Układ trójkolumnowy — stary sposób*

```
<table>
      <tr>
            <td colspan="2">
                  <p>górny pasek nawigacyjny (znak firmowy i reklamy)</p>
            </td>
      </tr>
      <tr>
            <td>
                  <p>lewy pasek nawigacyjny</p>
            </td>
           <td>
                  <p>główny obszar</p>
            </td>
      </tr>
</table>
```

W arkuszu CSS komórki tabeli są zastąpione elementami `div` (listing 8.10).

Listing 8.10. *Układ trójkolumnowy — nowy sposób*

```
<div id="adv">
      <p>górny pasek nawigacyjny (znak firmowy i reklamy)</p>
</div>
<div id="nav">
      <p>lewy pasek nawigacyjny</p>
</div>
<div id="main">
      <p>główny obszar</p>
</div>
```

Powyższy kod jest znacznie bardziej przejrzysty. Zwróć uwagę, że elementy `div` są nazywane funkcjonalnie, a nie na podstawie lokalizacji. Dodajmy teraz CSS, aby odpowiednio rozmieścić elementy (listing 8.11).

Listing 8.11. *Układ trójkolumnowy — CSS*

```
<style type="text/css">
<!--
div#adv {
      background:#ccc;
      width:auto;
      margin:0;
      padding:0.5em;
}
div#nav {
      background:#ddd;
      float:left;
      width:25%;
      margin-right:0.5em;
      padding:0.5em;
}
div#main {
      background:#0fc;
      margin-left:25%;
      padding:0.5em;
}
-->
</style>
```

Kluczowymi elementami dla układów dwu- i trójkolumnowych są: `float`, `margin-left` i `right`. `float` przesuwa dany element do prawej lub lewej strony (pozostała zawartość go „otacza”). W tym przypadku nawigacyjny `div` jest ustawiony po lewej stronie, ma określoną szerokość i margines wewnętrzny oraz margines zewnętrzny po stronie prawej. Dla głównej zawartości `div` ustawiany jest lewy margines zewnętrzny, który ma odpowiadać szerokości elementu przesuniętego do lewej strony.

Zwiększenie skuteczności

Im wyżej umieścisz główną zawartość dokumentu w kodzie HTML, tym większa będzie skuteczność wyszukiwania słów kluczowych na stronie. Można wykorzystać CSS do sztuczki wyświetlania tabeli opisanej w rozdziale 4. Na listingu 8.12 przedstawiony został sposób odwrotnego wyrównywania elementów — zamiast przesunięcia paska nawigacyjnego do lewej strony główna zawartość `div` jest przesuwana do strony prawej.

Listing 8.12. *Udoskonalony układ trójkolumnowy (kod CSS)*

```
<style type="text/css">
<!--
div#adv {
      background:#ccc;
      width:auto;
      margin:0;
      padding:0.5em;
}
div#main {
      float:right;
      width:75%;
      background:#0fc;
      margin-left:0.5em;
      padding:0.5em;
}
div#nav {
      background:#ddd;
      margin-right:75%;
      padding:0.5em;
}
-->
</style>
```

Można teraz zamienić pozycje `div`-ów `main` z `nav`, umieszczając główną zawartość wcześniej w kodzie HTML (listing 8.13).

Listing 8.13. *Udoskonalony układ trójkolumnowy (kod HTML)*

```
<div id="adv">
      <p>górny pasek nawigacyjny (znak firmowy i reklamy)</p>
</div>
<div id="main">
      <p>główna zawartość</p>
</div>
      <p>lewy pasek nawigacyjny</p>
</div>
```

Strona wygląda identycznie, lecz główna zawartość jest wcześniej umieszczona w kodzie, dzięki czemu jest szybciej wyświetlana. Jeżeli chcesz, możesz ją nawet umieścić na samej górze dokumentu, a dopiero za nią `div` z reklamami i nawigacją.

Dodatkowe informacje

Na poniższych stronach znajdziesz dodatkowe informacji o sposobach tworzenia układów dokumentów na podstawie CSS:

- *http://www.alistapart.com* — A List Apart korzysta z układu opartego na arkuszach CSS i uczy programistów, jak z nich korzystać.
- *http://www.meyerweb.com/eric/css/edge/* — strona *CSS Edge* Erica Meyera jest prawdziwym placem zabaw z metodami bazującymi na arkuszach CSS.
- *http://developer.apple.com/internet/css/introcsslayout.html* — wstęp do układów stron opartych na arkuszach CSS (autorstwa Erica Costello).

Podsumowanie

Aby w pełni wykorzystać arkusz CSS, niezbędne jest przekształcenie kodu dokumentu. Należy korzystać ze skróconych nazw `class` i `id`, używać skróconych właściwości oraz grupować właściwości minimalizując rozmiar arkusza stylów. Należy zamienić stary tabelaryczny układ dokumentu oraz nakładki oparte na obrazkach i JavaScripcie na strukturę bazującą na arkuszach CSS. Aby zmniejszyć tabele, należy nadać styl za pomocą arkuszy CSS (a nie umieszczać w każdym miejscu właściwości `bgcolor`). Warto także przenieść główną zawartość dokumentu na wyższą pozycję w kodzie HTML (sztuczka wyświetlania tabeli albo metody CSS). Poniżej przedstawiona została lista wskazówek omówionych w niniejszym rozdziale:

- Używaj skróconych właściwości, grupowania i skrótów do optymalizacji arkuszy CSS.
- Zamieniaj kod efektów nakładkowych JavaScript/obrazki na CSS.
- Twórz efekty nakładkowe CSS2 za pomocą pseudoklasy `hover`, stylizowanych łączy albo list.
- Używaj `table-layout:fixed`, `colgroup` i `col`, aby zwiększyć prędkość wyświetlania tabel.
- Używaj arkuszy CSS do ustawiania kolorów komórek tabeli (działa tylko z przeglądarkami w wersji 4+).
- Definiuj układ i rozmieszczenie elementów poprzez arkusze CSS i XHTML, aby oddzielić warstwę prezentacyjną od struktury i zmniejszyć rozmiar kodu.
- Staraj się, aby układ dokumentu był elastyczny.
- Zwiększ skuteczność wyszukiwania na swojej stronie.

Polecane publikacje

- „Cascading Style Sheets: The Definitive Guide", Eric Meyer, O'Reilly 2000r. („CSS. Kaskadowe arkusze stylów. Przewodnik encyklopedyczny", Helion, 2001 r.) — doskonałe wprowadzenie do CSS.
- „Eric Meyer on CSS", Eric Meyer, New Riders, 2002 r. — Meyer przedstawia prawdziwy świat CSS (13 projektów konwersji).
- A List Apart pod przewodnictwem Jefferya Zeldmana (*http://www.alistapart.com*) — dobre źródło informacji o standardach internetowych, w tym między innymi o CSS.
- Cascading Style Sheets 1, 2 i 3, World Wide Web Consortium (*http://www.w3.org/Style/CSS/*) — oficjalna specyfikacja CSS.
- The Web Standards Project pod przewodnictwem Jefferya Zeldmana (*http://www.webstandards.org*) — ustalenie standardów dla producentów przeglądarek internetowych. Powrót do WaSP (The Web Standards Project).