

Zawiera  
CSS3

# CSS

Wydanie III

Witryny internetowe  
szyte na miarę

Charles Wyke-Smith

AUTORYTETY INFORMATYKI

New  
Riders

Helion



Tytuł oryginału: Stylin' with CSS: A Designer's Guide, Third Edition

Tłumaczenie: Maksymilian Gutowski

na podstawie: „CSS. Witryny internetowe szyte na miarę. Autorytety informatyki. Wydanie II”  
w tłumaczeniu: Łukasza Piwko

ISBN: 978-83-246-7066-6

Authorized translation from the English language edition, entitled: STILIN' WITH CSS:  
A DESIGNER'S GUIDE, Third Edition; ISBN 0321858476; by Charles Wyke-Smith;  
published by Pearson Education, Inc, publishing as New Riders Publishing.

Copyright © 2013 by Charles Wyke-Smith.

All rights reserved. No part of this book may be reproduced or transmitted in any form  
or by any means, electronic or mechanical, including photocopying, recording or by  
any information storage retrieval system, without permission from Pearson Education, Inc.

Polish language edition published by HELION S.A. Copyright © 2013.

All rights reserved. No part of this book may be reproduced or transmitted in any form  
or by any means, electronic or mechanical, including photocopying, recording or by  
any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu  
niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą  
kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym  
lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź  
towarowymi ich właścicieli.

Autor oraz Wydawnictwo HELION dołożyli wszelkich starań, by zawarte w tej książce  
informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich  
wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich.  
Autor oraz Wydawnictwo HELION nie ponoszą również żadnej odpowiedzialności za ewentualne  
szkody wynikłe z wykorzystania informacji zawartych w książce.

Wydawnictwo HELION

ul. Kościuszki 1c, 44-100 GLIWICE

tel. 32 231 22 19, 32 230 98 63

e-mail: [helion@helion.pl](mailto:helion@helion.pl)

WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<http://helion.pl/user/opinie/csswi3>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

# Spis treści

Podziękowania • iii

O autorze • iv

Spis treści • v

Wstęp • x

## **ROZDZIAŁ 1: KOD HTML I STRUKTURA DOKUMENTU • 1**

Podstawy kodu HTML • 2

Znaczniki okalające — tekst • 2

Znaczniki nieokalające — treści wskazywane  
poprzez odniesienie • 3

Atrybuty • 4

Nagłówki i akapity • 5

Elementy złożone • 5

Zagnieżdżone znaczniki • 6

**Budowa dokumentu HTML • 7**

Szablon strony HTML • 7

Elementy blokowe i liniowe • 10

Elementy zagnieżdżone • 16

**Obiektowy model dokumentu (DOM) • 20**

Podsumowanie • 22

## **ROZDZIAŁ 2: PODSTAWY CSS • 23**

Budowa reguły CSS • 24

Konwencje zapisu reguł CSS • 26

**Selektory kontekstowe • 28**

**Wyspecjalizowane selektory kontekstowe • 32**

Selektor dziecka > • 32

Selektor sąsiadującego brata + • 33

Ogólny selektor braci ~ • 33

Selektor uniwersalny \* • 34

**Identyfikatory i klasy • 35**

Atrybut class • 35

Atrybut id • 38

Kiedy używać identyfikatorów, a kiedy klas? • 39

Identyfikatory i klasy — podsumowanie • 41

**Selektory atrybutów • 41**

Selektor nazwy atrybutu • 41

Selektor wartości atrybutu • 42

Selektory atrybutów — podsumowanie • 42

**Pseudoklasy • 43**

Pseudoklasy interfejsu • 43

Pseudoklasy strukturalne • 46

**Pseudoelementy • 47**

**Dziedziczenie • 49**

**Kaskadowość • 50**

Źródła stylów • 50

Zasady kaskadowości • 52

Obliczanie precyzji • 53

**Deklaracje reguł • 55**

Wartości słowne • 55

Wartości liczbowe • 56

Wartości kolorów • 57

**Podsumowanie • 61**

## **ROZDZIAŁ 3: POZYCJONOWANIE ELEMENTÓW • 62**

**Model połowy • 62**

Obramowanie • 63

Dopełnienie • 66

Margines • 67

Scalanie marginesów • 68

Wybieranie jednostek miary marginesów • 69

**Wielkość pola • 70**

**Elementy pływające i oczyszczające • 75**

Właściwość float • 76

Trzy sposoby włączania pływających elementów do kontenerów • 78

**Właściwość position • 86**

Pozycjonowanie statyczne • 86

Pozycjonowanie względne • 87

Pozycjonowanie bezwzględne	• 88
Pozycjonowanie stałe	• 89
Kontekst pozycjonowania	• 90
<b>Właściwość display</b>	<b>• 93</b>
<b>Tła</b>	<b>• 93</b>
Właściwości tła CSS	• 94
Kolor tła	• 95
Obraz tła	• 95
Powtórzenia obrazu tła	• 96
Położenie tła	• 97
Wielkość tła	• 99
Zaczeplenie tła	• 100
Właściwość zbiorcza tła	• 101
Inne właściwości tła w CSS3	• 101
Większa liczba obrazów tła	• 102
Gradyenty tła	• 104
<b>Podsumowanie</b>	<b>• 107</b>

## **ROZDZIAŁ 4: STYLIZOWANIE FONTÓW I FORMATOWANIE TEKSTU • 108**

<b>Fonty</b>	<b>• 108</b>
Właściwość font-family	• 109
Właściwość font-size	• 112
Właściwość font-style	• 115
Właściwość font-weight	• 116
Właściwość font-variant	• 116
Właściwość font	• 117
<b>Właściwości tekstu</b>	<b>• 117</b>
Właściwość text-indent	• 118
Właściwość letter-spacing	• 119
Właściwość word-spacing	• 121
Właściwość text-decoration	• 122
Właściwość text-align	• 122
Właściwość line-height	• 123
Właściwość text-transform	• 124

Właściwość vertical-align • 125

**Fonty internetowe • 126**

Internetowe biblioteki fontów • 127

Gotowe zestawy @font-face • 128

Własne zestawy @font-face • 130

**Stylizacja tekstu • 130**

Podstawowy układ tekstu • 131

Stylizowanie tekstu w siatce • 135

Typografia klasyczna • 141

**Podsumowanie • 150**

## **ROZDZIAŁ 5: LAYOUTY • 151**

**Podstawy tworzenia layoutów • 151**

Wysokość i szerokość layoutu • 152

**Tworzenie kolumn • 153**

Nadawanie kolumnom dopełnień i obramowań • 161

**Trzykolumnowe layouty z płynną środkową kolumną • 172**

Trzykolumnowy layout z płynną środkową kolumną i ujemnymi marginesami • 172

Trzykolumnowy layout z płynną środkową kolumną, oparty na właściwościach CSS3 table • 177

**Layout wielorzędowy i wielokolumnowy • 179**

Praktyczne selektory CSS • 182

Wewnętrzne elementy div w działaniu • 184

**Podsumowanie • 185**

## **ROZDZIAŁ 6: KOMPONENTY INTERFEJSU • 186**

**Tworzenie menu nawigacyjnych • 186**

Pionowe menu • 186

Menu poziome • 189

Rozwijane menu • 191

**Formularze • 201**

Elementy HTML formularza • 201

Sposoby kodowania formularzy • 209

Stylizacja formularza • 210

Formularz wyszukiwania • 221

- Chmurka • 224
- Stosy i z-index • 227
- Tworzenie trójkąta w CSS • 228
- Podsumowanie • 230

## **ROZDZIAŁ 7: STRONA INTERNETOWA Z CSS3 • 231**

- Struktura strony • 231
- Planowanie kodu HTML • 232
- Stylizacja nagłówka • 236
- Obszar tytułowy • 237
- Formularz wyszukiwania • 239
- Menu • 242
- Obszar treści • 249
- Stylizacja pola logowania • 253
- Odnosniki do wpisów • 258
- Obszar książek • 260
- Stopka • 268
- Podsumowanie • 271

## **ROZDZIAŁ 8: PROJEKTOWANIE SKALOWALNE • 272**

- Duże layouty na małych urządzeniach • 272
- Zapytania medialne • 274
- Reguła @media • 274
- Atrybut media znacznika link • 277
- Wartości graniczne • 277
- Wartość viewport znacznika meta • 278
- Optymalizacja layoutu na potrzeby tabletów • 278
- Optymalizacja layoutu dla smartfonów • 282
- Dostosowanie layoutu do orientacji pionowej • 285
- Ostatnie detale • 287
- Błąd ze skalowaniem w Safari Mobile • 287
- Rozwijane menu na ekranach dotykowych • 287
- Podsumowanie • 290

## **DODATEK • 291**

## **SKOROWIDZ • 299**

# Strona internetowa z CSS3

**W TYM ROZDZIALE WYKORZYSTAM** techniki tworzenia layoutów z rozdziału 5. oraz techniki stylizacji komponentów z rozdziału 6., by stworzyć pełnoprawną witrynę. Zapoznasz się z wieloma funkcjami stylistycznymi CSS3, m.in. zaokrąglonymi rogami, cieniami tekstu i pól, przejściami oraz przekształceniami, które nadają oprawie graficznej bardziej nowoczesny i profesjonalny wygląd.

Witryna, którą stworzę w tym rozdziale, to ta, którą tworzę równoległe z tą książką — jej nowa strona internetowa. Tak jak zawsze, zaprezentuję na tym przykładzie nowe techniki, z których będziesz mógł korzystać we własnych projektach, oraz zwrócę uwagę na utrudnienia, z którymi możesz się zetknąć.

Pokażę Ci najpierw, jak rozplanować szkielet strony, a następnie przeprowadzę krok po kroku przez proces nadawania stylów CSS każdemu obszarowi strony. Po przeczytaniu tego rozdziału będziesz wiedział, czego potrzeba do stworzenia pełnej witryny, i będziesz gotów tworzyć własne.

## Struktura strony

Kiedy tworzysz jakąkolwiek bardziej skomplikowaną stronę, musisz napisać setki linijek kodu HTML i CSS. Warto zatem, żeby był uporządkowany. Ważne jest, aby nadać kodowi logiczną strukturę i myśleć hierarchicznie, żeby porządek kodu CSS był zgodny z kodem HTML. Tak właśnie sformatowany jest kod kolejnego przykładu i zdecydowanie polecam przyjęcie takiego podejścia. Wymaga to pewnej dyscypliny, ale zdecydowanie się opłaca, ponieważ łatwo dzięki temu znaleźć kod CSS odnoszący się do dowolnego fragmentu kodu HTML. Ponadto możesz dzięki temu uniknąć dezorientującej sytu-



acji, w której kilka rozrzuconych po kodzie reguł nadaje style jednemu elementowi. To podejście przede wszystkim jednak ułatwia zrozumienie kodu i jego edycję, zarówno przeze mnie, jak i przez innych.

Rzućmy najpierw okiem na ukończoną witrynę (rysunek 7.1).



RYSUNEK 7.1. Oto ukończona witryna

## Planowanie kodu HTML

Kiedy dopiero zaczynasz pracę z HTML, przełożenie projektu graficznego na elementy kodu może być nie lada wyzwaniem. Można jednak przyjąć pewne dobre podejście. Przed rozpoczęciem kodowania strony sam projekt zwykle tworzy się w Photoshopie, Fireworks lub chociaż na kartce papieru. To na tym etapie możesz omówić projekt z klientem, a następnie dostosować go, by mieć pewność, że oprawa graficzna i organizacja treści zgadzają się z wymogami projektu. Następnie należy zabrać się za kodowanie stron. Pierwszym krokiem jest rozrysowanie w layoucie pól przedstawiających główny, strukturalny kod HTML.

Ponieważ kod HTML tworzy prostokątne elementy, powinieneś znaleźć sposób, by podzielić layout na kilka możliwie największych obszarów, określających najwyższe poziomy kodu, a następnie podzielić je prostokątnymi polami na mniejsze sekcje, zawierające zstępne elementy strukturalne. Oto, jak ten podział wygląda na mojej stronie.



RYСУNEK 7.2. Struktura pól na stronie pozwala mi wyłonić trzy poziomy strukturalne

Jak widać na **rysunku 7.2**, strona podzielona jest schludnie na cztery prostokątne (pomarańczowe) pola, zajmujące całą szerokość strony, które odnoszą się do elementów HTML najwyższego poziomu. Zauważ, że zamierzam także zawrzeć cały layout w jednym kontenerze (oznaczonym tu na zielono), aby móc z łatwością ustawić ogólną szerokość layoutu i wyśrodkować treść w oknie przeglądarki. Warto na tym etapie również wybrać klasy i identyfikatory głównych znaczników. Elementy HTML **header** i **footer** nie potrzebują klasy ani identyfikatora, gdyż w dokumencie występują tylko raz, ale dwa środkowe prostokąty — elementy **section** — trzeba będzie odróżnić identyfikatorami: **feature\_area** i **book\_area**.



Fakt, że obszar nawigacji pokrywa się z obszarem tytułu i pola wyszukiwania, wynika z tego, że już postanowiłem, iż te dwa ostatnie obszary będą pozycjonowane bezwzględnie. Sprawia to, że element `nav` ignoruje je i wypełnia cały nagłówek. Będę mógł następnie wyśrodkować zawarte w nim menu na stronie, co zademonstruję w dalszej części rozdziału.

Kolejnym krokiem jest rozplanowanie struktury drugiego poziomu, która oznaczona tu jest niebieskimi prostokątami. Przyglądam się każdemu elementowi najwyższego poziomu, aby określić największe elementy, w jakich można zorganizować ich zawartość.

W nagłówku `header` znajdują się trzy zbiory treści: obszar tytułu po lewej, menu nawigacyjne pośrodku oraz formularz wyszukiwania po prawej. W obszarze `feature_area` wpis blogowy umieściłem po lewej, a po prawej obszar `aside` z polem logowania i listą odnośników do wpisów. Zauważ, że w jednym elemencie zawarłem *obydwa* te komponenty; obszar ten podzielę dalej w kolejnym kroku.

W obszarze `book_area` znajdują się cztery kontenery na cztery książki. W znaczniku `footer` znajduje się tekst oraz strukturalny element `nav`.

Muszę następnie dalej podzielić obszar `feature_area`, aby uzyskać dwa osobne komponenty: formularz i odnośniki. Wszystkie obrazy książek w obszarze `book_area article` zawarłem we własnych elementach, aby uzyskać kontekst pozycjonowania dla chmurki informacyjnej każdej z książek. Te dodatkowe elementy składają się na trzeci poziom hierarchii strukturalnej i oznaczone są fioletowymi prostokątami.

Tego planowania już wystarczy, by zacząć pisać kod, choć w ramach tworzenia strony konieczne może się okazać dodanie jednego czy dwóch elementów. Utworzę teraz wstępny kod, który odzwierciedla opisaną strukturę.

```
<div id="wrapper">
```

poziom pierwszy		<code>&lt;header&gt;</code>
poziom drugi		<code>&lt;section id="title"&gt;</code> <code>&lt;!-- nagłówki h1 i h2 --&gt;</code> <code>&lt;/section&gt;</code>
poziom drugi		<code>&lt;nav class="menu"&gt;</code> <code>&lt;!-- menu nawigacyjne --&gt;</code> <code>&lt;/nav&gt;</code>
poziom drugi		<code>&lt;form class="search"&gt;</code> <code>&lt;!-- pole wyszukiwania --&gt;</code>

```

        </form>
    </header>
    poziom pierwszy | <section id="feature_area">
    poziom drugi   | <article id="blog_leadoff">
                   | <!-- treść bloga -->
                   | </article>
    poziom drugi   | <aside>
    poziom trzeci  | <form class="signin">
                   | <!-- pole logowania -->
                   | </form>
    poziom trzeci  | <nav>
                   | <!-- odnośniki do wpisów -->
                   | </nav>
                   | </aside>
    </section>
    poziom pierwszy | <section id="book_area">
    poziom drugi   | <article>
    poziom trzeci  | <div class="inner">
                   | <!-- obrazy książek i obrócony tekst -->
                   | </div>
                   | </article>
                   | <!-- cztery elementy article -->
    </section>
    poziom pierwszy | <footer>
                   | <!-- tekst elementów footer i nav -->
                   | </footer>
    poziom trzeci  | </div>

```

Jak widzisz, każdy poziom osadzonych pól layoutu odzwierciedlony jest w kodzie w postaci osadzonych elementów. Najbardziej odpowiednie elementy HTML można dobierać na bieżąco. Lista menu na przykład zdecydowanie powinna znaleźć się w elemencie `nav`.

Mając już ukończony kod strukturalny, określam ogólne ustawienia fonta i koloru tła strony dla elementu `body` oraz definiuję styl kontenera `wrapper`, by podać ogólną szerokość layoutu i wyśrodkować go na stronie.

```
body {
    font-family:helvetica, arial, sans-serif;
    background:#efefef;
    margin:0;
}
wrapper {width:980px; margin:0 auto 20px;}
```

Mogę teraz zająć się kolejnymi elementami na stronie, nadając im treść i style CSS.

## Stylizacja nagłówka

Zacznijmy od utworzenia kodu treści nagłówka `header`.

```
<header>
  <section id="title">
    <h1>Strona o CSS</h1>
    <h2>Blog i książki Charlesa Wyke-Smitha</h2>
  </section>
  <nav class="menu">
    <ul>
      <li class="choice1"><a href="#">Artykuły</a></li>
      <li class="choice2"><a href="#">Książki</a></li>
      <li class="choice3"><a href="#">Materiały</a></li>
      <li class="choice4"><a href="#">Biblioteczka</a></li>
      <li class="choice5"><a href="#">Kontakt</a></li>
    </ul>
  </nav>
```

atrybut `for` (o takiej samej wartości, jak identyfikator kontrolki) łączy oznaczenie z kontrolką

```
<form class="search" action="#" method="post">
  <label for="search">search</label>
  <input type="text" id="search" name="search"
    placeholder="szukaj" />
</form>
</header>
```

Znacznik `header` dzieli się na trzy części: tytuł, obszar wyszukiwania i wyśrodkowane menu. Zacznę od oznaczonego identyfikatorem `title` obszaru tytułowego.

## Obszar tytułowy

Elementy `h1` i `h2` pozycjonują bezwzględnie w górnym lewym rogu elementu `header`. Oto kod CSS:

```
header {
  position:relative;
  height:70px;
  margin:10px 0;
  background:#fff;
  border-radius:20px 0px 20px 0px;
  box-shadow:0 12px 8px -9px #555;
  padding:1px;
}

header section#title {
  position:absolute;
  width:300px;
  height:65px;
  left:0px;
```

kontekst pozycjonowania obszaru tytułowego i wyszukiwania

ustalona wysokość elementu obejmującego elementy pozycjonowane bezwzględnie

kolejność: lewy górny róg, prawy górny, prawy dolny, lewy dolny

ujemne oddalenie sprawia, że cień nie przekracza szerokości pola

spawia, że marginesy elementu i jego dzieci nie ulegają scaleniu

odpowiednio duża szerokość pozwala zapobiec zawijaniu tekstu

wystarczająco duża wysokość do objęcia dwóch wierszy tekstu

pozycjonowanie w lewym górnym rogu

```

        top:0;
    }
    header h1 {
        padding:9px 12px 0;
        font-family:'Lato', helvetica, sans-serif;
        font-weight:900;
        font-size:2.2em;
        line-height:1;
        letter-spacing:-.025em;
        color:#4eb8ea;
    }
    header h2 {
        padding:0px 12px;
        font-family:"Source Sans Pro", helvetica, sans-serif;
        font-weight:400;
        font-size:.9em; line-height:1;
        letter-spacing:-.025em;
        color:#333;
    }

```

grubość, którą trzeba określić dla pobieranych fontów

## Strona o CSS

Blog i książki Charlesa Wyke-Smitha

RYSUNEK 7.3. Elementy h1 i h2 są już wystylizowane. Tymczasowo włączone obramowania ukazują położenie elementów

Pierwsze, na co warto zwrócić uwagę w tym kodzie, to ustalona wysokość elementu `header`. Jak pokazałem w wielu poprzednich przykładach, zwykle warto pozwolić, by treść określała wysokość elementów strukturalnych, a wysokość strony zmieniała się wraz z dodawaniem treści. W tym przypadku, jako że `header` zawiera pozycjonowane bezwzględnie elementy, które nie rozpychają swoich rodziców, wysokość musiałem określić sam. Zawartość elementu `header` nie będzie ulegać większym zmianom, jeśli w ogóle, więc jest mało prawdopodobne, że kiedykolwiek wyjdzie poza obszar zdefiniowany jego wysokością.

Zwróć uwagę na ciekawe zestawienie dwóch zaokrąglonych i dwóch zwykłych rogów, które nadałem elementowi `header` oraz wielu innym znacznikom na stronie. Ten dyskretny a wyrazisty efekt dodaje layoutowi niepowtarzalnego charakteru. W ramce „Zaokrąglone rogi” znajdziesz więcej szczegółów związanych z tworzeniem zaokrąglonych rogów pól elementów HTML.

Pozycję elementu `title` (oznaczonego kolorem czerwonym na **ry-sunku 7.3**) określiłem bezwzględnie, a następnie zdefiniowałem wielkość i dopełnienie jego elementów tekstowych oraz ustaliłem jego wysokość i szerokość, żeby te elementy obejmował.

Zauważ, że w tych nagłówkach używam fonta internetowego Lato z Google Web Fonts. Grubość wielu fontów internetowych określa się właściwością `font-weight`, inaczej niż w przypadku fontów zainstalowanych w systemie, gdzie dla tej właściwości można określić jedynie styl normalny i pogrubiony. Więcej na temat Google Web Fonts przeczytasz w rozdziale 4.

Cienie pól to kolejny charakterystyczny aspekt tego projektu. Są one zdefiniowane tak, by pojawiały się tylko przy dolnej krawędzi pola i miały od pola mniejszą szerokość. Tworzy to wrażenie, że pole unosi się nad stroną. Przykład znajdziesz w podświetlonym kodzie w powyższym przykładzie, a więcej na temat cieni pól przeczytasz w ramce „Cienie pól”.

Mając już gotowe pole tytułowe, w podobny sposób umieścimy pole wyszukiwania po lewej stronie.

## Formularz wyszukiwania

Zacznijmy od kodu:

```
<form class="search" action="#" method="post">
  <label for="search">search</label>
  <input type="text" id="search" name="search"
    placeholder="szukaj" />
</form>
```

Poniżej znajduje się kod CSS formularza, który pojawił się w przykładzie w poprzednim rozdziale. Jedyna istotna różnica dotyczy tego, w jaki sposób określone jest położenie formularza w nagłówku.



## Zaokrąglone rogi

Tworzenie zaokrąglonych rogów, które były charakterystyczne dla designu Web 2.0 kilka lat temu, wymagało korzystania z rozbudowanego kodu JavaScript lub starannego rozmieszczania plików graficznych w zagnieżdżonych elementach `div`. Dziś wystarczy napisać jedną linię kodu CSS.

Podstawowa składnia wygląda następująco:

```
border-radius:10px;
```

W tym przypadku zaokrąglenie wszystkich czterech rogów ma wielkość 10 pikseli.

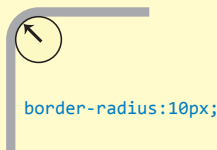
Można tutaj użyć standardowych deklaracji zbiorczych pola, z tym że nie obowiązuje kolejność właściwości `top`, `right`, `bottom`, `left` (które odnoszą się do krawędzi), lecz `top-left`, `top-right`, `bottom-right`, `bottom-left` (które odnoszą się do rogów).

Zauważ, że możesz określić zarówno poziomy, jak i pionowy promień zaokrąglenia

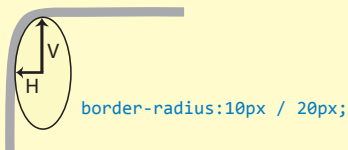
```
border-radius:10px / 20px;
```

Na rysunku 7.4 widać, że podane wartości określają promień koła lub elipsy, na której oparte jest zaokrąglenie.

```
border-radius:10px;
```



```
border-radius:10px;
```



```
border-radius:10px / 20px;
```

RYSUNEK 7.4. Ten diagram przedstawia rezultaty zastosowania dwóch powyższych przykładów kodu

Jeżeli chcesz nadać odmienne wartości promieni poziomym i pionowym każdemu rogowi, możesz napisać deklarację zbiorczą:

```
border-radius:10px 6px 4px 12px / 20px 12px 8px 24px; /* cztery poziome promienie, cztery pionowe */
```

Zauważ, że obramowanie nie musi być widoczne, by korzystać z zaokrąglonych rogów. Jak widać na przykładzie menu z tego rozdziału, kolor tła elementu wyświetla samo zaokrąglenie bez obramowania.

szerokość pozwalająca na objęcie poszerzonego pola	—	<code>form.search {</code>
		<code>position:absolute; width:150px;</code>
położenie określone względem prawego górnego rogu nagłówka	—	<code>top:23px; right:20px;</code>
		<code>}</code>
		<code>.search input {</code>
tworzy miejsce, w którym pole może się poszerzyć w lewo	—	<code>float:right; width:70px;</code>
		<code>padding:2px 0 3px 5px;</code>
		<code>border-radius:10px 0px 10px 0px;</code>
		<code>font-family:"Source Sans Pro", helvetica, sans-serif;</code>

## Cienie pól

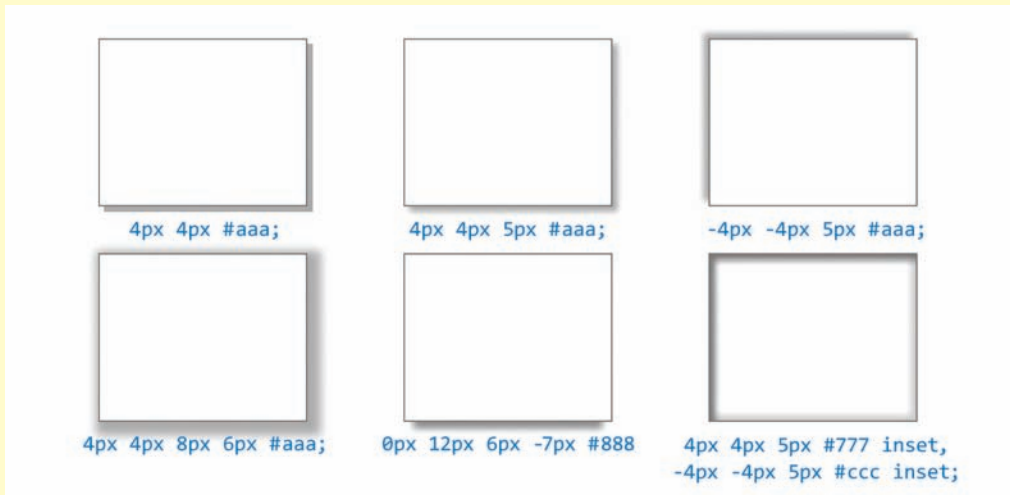
Cienie pól elementów HTML są kolejnym przykładem efektu, który — zanim pojawił się CSS3 — uzyskiwano przy użyciu różnych elementów graficznych, elementów `div`, co wymagało niemałej cierpliwości, a dziś można uzyskać dzięki jednej linijce kodu CSS.

Składnia podstawowej deklaracji wygląda następująco:

```
box-shadow:4px 4px 5px 8px #aaa inset;
```

Kolejność stylów jest następująca: przesunięcie w poziomie, przesunięcie w pionie, rozmycie, rozszerzenie, kolor, wypuszczenie cienia do wewnątrz pola (cień znajduje się domyślnie poza polem).

Musisz przynajmniej podać przesunięcie w poziomie i w pionie oraz kolor — uzyskasz wtedy cień o wyraźnej krawędzi, o podanej szerokości i barwie. Nadając wartości ujemne ustawieniom przesunięcia w prawo i w dół, przesuwasz cień w lewo i w górę. Słowo kluczowe `inset` wpuszcza cień w pole. Można podać więcej niż jedną deklarację cienia, oddzielając je od siebie przecinkami. Na rysunku 7.5 widzimy różne przykłady tego, co można tą właściwością uzyskać.



RYSUNEK 7.5. Przy użyciu różnych wartości, zarówno dodatnich, jak i ujemnych, można uzyskać różne efekty cienia pola

```
font-weight:400;
font-size:1em;
color:#888;
outline:none;
-webkit-transition:2s width;
}
.search input:focus {width:140px;}
```

usuwa domyślne podświetlenie obramowania —

tworzy animację zmiany wielkości pola (wymagane są tu także inne prefiksy) —

element rozciąga się na taką szerokość po sfokusowaniu —

```
.search label {display:none;}
form.search input {background-color:#fff;}
form.search input::-webkit-input-placeholder {color:#ccc;}
```

RYSUNEK 7.6. Przejście zwiększa szerokość pola. Tymczasowo widać obramowanie formularza

Jak widać w wyróżnionym kodzie, określam szerokość elementu `form` i określam położenie jego prawej krawędzi względem prawej krawędzi elementu `header`. Jak wskazuje tymczasowo widoczne obramowanie na **rysunku 7.6**, kontener formularza jest wystarczająco duży, by obejmować rozszerzone pole. Nie będę się rozwodzić nad samym obszarem wyszukiwania, ponieważ omówiłem go już w poprzednim rozdziale, ale wspomnę, że tekst zastępczy — przynajmniej w przeglądarkach opartych na Webkit — można wystylizować inaczej niż tekst, który użytkownik podaje w polu, co widać w ostatniej linijce powyższego kodu.

## Menu



W celu zaoszczędzenia miejsca nie pokazuję osadzonych poziomów listy, które składają się na rozwijane menu. O tworzeniu takich menu przeczytasz w punkcie „Rozwijane menu” w rozdziale 6.

Rozmieściłem już elementy bezwzględnie przy obydwu krawędziach nagłówka, usuwając je ze struktury dokumentu. Mogę teraz umieścić między nimi wyśrodkowane menu. Oparte jest ono na kodzie CSS menu z rozdziału 6., z wyjątkiem kilku drobnych zmian.

```
<nav class="menu">
  <ul>
    <li class="choice1"><a href="#">Artykuły</a></li>
    <li class="choice2"><a href="#">Książki</a></li>
    <!-- kolejne elementy menu -->
  </ul>
</nav>
```

Kod składa się ze standardowej listy odnośników w elemencie `nav`, ale poszczególne odnośniki przypisałem do osobnych klas, aby nadać im różne kolory.

Oto kod CSS:

wyrównuje menu względem kontenera	<pre> nav.menu {     margin:19px auto;     padding:0;     text-align:center;     font-size:.8em; } </pre>
kontener ściśle obejmuje elementy listy	<pre> nav.menu &gt; ul {display:inline-block;} nav.menu li { </pre>
rozkłada menu w poziomie	<pre> float:left; </pre>
usuwa domyślne punktory listy	<pre> list-style-type:none; </pre>
kontekst pozycjonowania osadzonej listy	<pre> position:relative; } </pre>
sprawia, że odnośnik wypełnia element li	<pre> nav.menu li a {     display:block;     padding:.25em .8em;     font-family:"Source Sans Pro", helvetica, sans-serif;     font-weight:600;     font-size:1.2em;     text-align:left;     color:#fff; </pre>
usuwa podkreślenie odnośnika	<pre> text-decoration:none; </pre>
zapobiega migotaniu tekstu pod koniec zmiany przezroczystości w przeglądarkach opartych na Webkit	<pre> -webkit-font-smoothing:antialiased; } nav.menu li.choice1 a {background:#f58c21;} nav.menu li.choice2 a {background:#4eb8ea;} </pre>

```

nav.menu li.choice3 a {background:#d6e636;}
nav.menu li.choice4 a {background:#ee4c98;}
nav.menu li.choice5 a {background:#f58c21;}
nav.menu li:hover > a {
    color:#555;
    border-color:#fff;
    border:0;
}
nav.menu li:last-child a {border-bottom-right-radius:10px;}
nav.menu li:first-child a {border-top-left-radius:10px;}

```



RYSUNEK 7.7. Menu jest teraz wyśrodkowane na stronie. Pozycjonowane bezwzględnie elementy tytułu i pola wyszukiwania są wyjęte poza strukturalny ciąg dokumentu, wobec czego element `nav` może być szeroki na całą stronę (co wskazuje tymczasowo widoczne obramowanie)

Jako że obszary tytułu i wyszukiwania są pozycjonowane bezwzględnie, są one wyjęte z ciągu strukturalnego dokumentu. Element blokowy `nav` zachowuje się przez to tak, jakby nie były one obecne, i rozciąga się na całą szerokość swojego rodzica, `header` (rysunek 7.7). Mogę dzięki temu wyśrodkować menu na stronie. Posłużę się tym przykładem, aby bardziej szczegółowo omówić wyśrodkowywanie przy użyciu kodu CSS.

### WYŚRODKOWYWANIE ELEMENTÓW O NIEOKREŚLONEJ SZEROKOŚCI

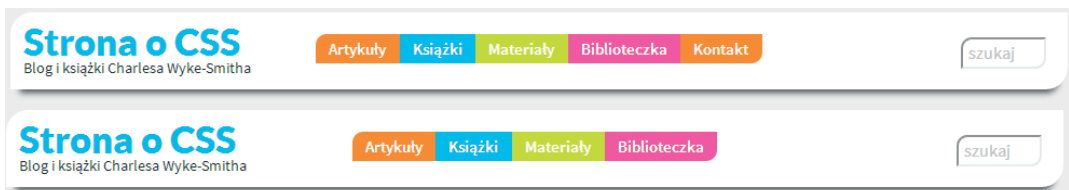
Wyśrodkowanie elementu zawartego w drugim elemencie może być trudne. Kiedy masz do czynienia z elementami pozycjonowanymi standardowo wartością `static`, możesz sprawić, by element spływał w lewo lub w prawo, albo użyć właściwości `text-align` z rodzicem, aby wyrównać element do lewej (`left`), prawej (`right`) bądź wyśrodkować go (`center`). Element można także wyśrodkować przy użyciu marginesów o wartości `auto`. Minusem tych technik jest konieczność ustalenia szerokości elementu, który chce się tak rozmieścić. Kiedy lista tworząca takie menu generowana jest dynamicznie na

podstawie bazy danych, czy nawet jeśli jej elementy będą modyfikowane ręcznie, to nie sposób przewidzieć szerokości wyśrodkowanego elementu i ją ustalić. Często otrzymuję e-maile z pytaniami, jak wyśrodkowywać menu w kontenerze, wobec czego omówię wyśrodkowywanie elementów o nieokreślonej szerokości.

Wartość `inline-block` jest dziwnym, hybrydowym ustawieniem właściwości `display`, które łączy w sobie zachowanie elementów liniowych i blokowych. Zachowanie blokowe elementu o takim ustawieniu polega na obsłudze marginesów i dopełnień oraz możliwości obejmowania innych elementów blokowych. Zachowanie liniowe z kolei polega na ścisłym okalaniu zawartości zamiast rozszerzania się na całą szerokość rodzica. Oznacza to, że szerokość elementu zawsze jest równa szerokości jego treści. Kolejną użyteczną cechą wartości `inline-block` jest to, że element obejmuje pływające elementy. Problem w tym, że taki element *nie obsługuje* jednej wartości marginesu — `auto` — która jest najprostszym sposobem na wyśrodkowanie elementu w kontenerze.

Rozwiązaniem jest nadanie właściwości `text-align:center` rodzicowi elementu, czyli `nav`, a następnie nadanie wyśrodkowanemu elementowi właściwości `display:inline-block` — w tym przypadku elementu `ul` z elementami menu. To zestawienie pozwala na uzyskanie pożądanego rezultatu, czyli wyśrodkowanego w rodzicu elementu o nieokreślonej szerokości. Jak widać w pierwszych dwóch wyróżnionych wierszach podanego wcześniej kodu, tak właśnie zrobiłem. Menu jest teraz idealnie wyśrodkowane, gdyż rodzic `nav` ignoruje bezwzględnie pozycjonowane elementy znajdujące się po jego bokach, oraz rozciąga się na całą szerokość elementu `header`.

Aby zademonstrować, jak to się świetnie sprawdza, usunę jeden z elementów listy menu.



RYSUNEK 7.8. Menu pozostaje wyśrodkowane nawet po zmianie liczby zawartych w nim elementów

Na **rysunku 7.8** usunąłem ostatni element menu, które pozostaje idealnie wyśrodkowane. Jest to idealne rozwiązanie dla witryny z dynamicznie generowaną treścią, na której różni użytkownicy (np. zarejestrowani i niezarejestrowani) otrzymują różne menu. Zauważ też, że zaokrąglonych rogów nie nadałem pierwszemu i piątemu elementowi menu, gdyż uniemożliwiłoby to wprowadzanie zmian w menu. W celu uzyskania bardziej praktycznego kodu CSS, jak widać w wyróżnionym kodzie, zaokrąglone rogi nadałem elementom `:first-child` i `:last-child`. Po usunięciu piątej pozycji menu czwarty element staje się ostatnim i otrzymuje zaokrąglony róg określony selektorem `:last-child`.

### DODANIE ROZWIJANEGO MENU

W ramach dodania rozwijanego menu zaprezentuję kolejny przykład przejść w CSS.

ukrywa menu	<code>nav.menu li ul {</code>	<code>opacity:0; visibility:hidden;</code>
określa położenie względem menu będącego rodzicem		<code>position:absolute;</code>
szerokość rozwijanego menu		<code>width:12em;</code>
wyrównuje lewą krawędź podmenu w stosunku do rodzica		<code>left:0;</code>
wyrównuje element w stosunku do dolnej krawędzi rodzica		<code>top:100%;</code>
tworzy przejście		<code>-webkit-transition:1s all;</code> <code>-moz-transition:1s all;</code> <code>transition:1s all;</code>
		<code>}</code>
	<code>nav.menu li:hover &gt; ul {</code>	<code>opacity:1; visibility:visible;</code>
obydwie właściwości podlegają przejściu		<code>}</code>
	<code>nav.menu li li {</code>	<code>float:none;</code>
niweluje odziedziczone pływanie elementów — sprawia, że odnośniki rozmieszczane są w pionie		<code>}</code>
		<code>nav.menu li li:first-child a {border-radius:0;}</code>
		<code>nav.menu li li:last-child a {border-bottom-left-radius:10px;}</code>
		<i>/* kod dla przeglądarek, które nie obsługują przejść CSS */</i>

przesłania wersję z przejściem	—	<code>.no-csstransitions nav.menu li ul {</code>
		<code>visibility:visible;</code>
przesłania wersję z przejściem	—	<code>opacity:1;</code>
ukrywa menu, jeśli przeglądarka nie obsługuje przejść CSS	—	<code>display:none;</code>
		<code>}</code>
wyświetla menu, kiedy kursor znajduje się nad rodzicem	—	<code>.no-csstransitions nav.menu li:hover &gt; ul {display:block;}</code>



RYSUNEK 7.9. Rozwijane menu dodane do menu głównego

Rozwijane menu tworzę takim samym kodem, jak w rozdziale 6. Żeby nie tłumaczyć wszystkiego od nowa, odsyłam Cię niniejszym do tego rozdziału i komentarzy do powyższego listingu. Chciałbym jednak zwrócić uwagę na trzy rzeczy obecne w tej wersji kodu.

Po pierwsze, zaokrąglone rogi są odziedziczone przez elementy rozwijanego menu. Zamiast zaokrąglić przeciwległe rogi, tak jak w menu głównym, chcę, by zaokrąglone były jedynie dolne rogi menu. Usuвам zatem odziedziczone zaokrąglenie w lewym górnym rogu i tworzę zaokrąglenie lewego dolnego rogu; zaokrąglenie prawego dolnego rogu pozostaje odziedziczone (rysunek 7.9).

Jak wskazują komentarze, przezroczystość menu ulega przejściu, żeby stawało się widoczne stopniowo. Za pierwszym podejściem nadałem jedynie właściwość `opacity` o początkowej wartości `0` (przezroczystość) i końcowej wartości `1` (pełna widoczność) elementowi, na którym znajduje się kursor. Dzięki temu menu rzeczywiście pojawiało się i znikало stopniowo, ale zawsze znajdowało się na swoim miejscu, nawet gdy nie było widoczne. Gdybym przesunął kursor w miejsce pod menu, rozwijane menu i tak się pojawiało, nawet jeśli kursor nie znajdował się nad odpowiednią pozycją w menu głównym. Spróbowałem następnie dodać właściwości `display:none` i `display:block`, aby całkowicie usunąć rozwijane menu, kiedy kursor nie znajdował się nad elementem menu. Wprawdzie rozwią-



załem w ten sposób problem, ale przez to rozwijane menu włączały się i wyłączały bez przejścia. Postanowiłem usunąć właściwość `display` i zamiast tego wyłączyć po najechaniu na element kursorem właściwość `visibility` podczas zmiany przezroczystości w ramach przejścia. W rezultacie rozwijane menu pojawiało się stopniowo, ale znikało bez przejścia. Wreszcie, po utworzeniu przejścia obejmującego zarówno `opacity`, jak i `visibility`, menu całkowicie znikało, gdy było przezroczyste, a zaczęło się pojawiać i znikać zgodnie z zamierzeniem, czyli stopniowo. Chcę zwyczajnie powiedzieć, że czasami trzeba poeksperymentować, żeby osiągnąć pożądany efekt. Mam nadzieję, że kiedyś oszczędzisz sobie dzięki temu kilku godzin pracy. Zawsze do usług.

Po trzecie, skorzystałem z okazji, by pokazać skrypt Modernizr w działaniu i podać zapasowy kod CSS odpowiedzialny za działanie menu w przeglądarkach, które nie obsługują przejść CSS3. W takim przypadku Modernizr nadaje klasę `no-csstransitions` elementowi głównemu `html` podczas wczytywania strony. Używam tej klasy w selektorach reguł, których mają używać jedynie te przeglądarki, które nie obsługują przejść CSS. W takich regułach anuluję ustawienia `visibility` i `opacity`, które określają działanie menu w przeglądarkach obsługujących przejścia, podając do wyświetlenia i ukrywania menu podstawowe reguły `display:none` i `display:block`, których użyłem przy menu w rozdziale 6.

Tym samym ukończyłem stylizację elementu `header`. Przejdźmy teraz do stylizacji obszaru obejmującego wpis blogowy, pole logowania i odnośniki do wpisów.

### Wyśrodkowywanie w pionie

Poziome wyśrodkowywanie w CSS nie jest łatwe. Kiedy wyśrodkowujesz pojedynczy wiersz tekstu w obrębie elementu o stałej wysokości, np. 300 pikseli, właściwość `line-height` tekstu nadaj wartość równą wysokości kontenera:

```
text-align:center; /* wyśrodkowywanie w poziomie */
```

```
line-height:300px; /* wyśrodkowywanie w pionie = wysokość kontenera */
```

Aby wyśrodkować pionowo pozostałe elementy, takie jak obrazy, właściwość `display` kontenera nadaj wartość `table-row`, a następnie nadaj mu działającą tylko z komórkami tabel właściwość `vertical-align` o wartości `middle`.

```
display:table-cell; /* uaktywnia zachowanie elementu jako tabeli */
```

```
vertical-align:middle; /* wyśrodkowywanie w pionie */
```

```
text-align:center; /* wyśrodkowywanie w poziomie */
```

Żadne z tych rozwiązań nie jest szczególnie eleganckie, ale w CSS nie ma konkretnych właściwości, które pozwalałyby na pozycjonowanie elementów w pionie.

## Obszar treści

W głównym obszarze treści znajdzie się wprowadzenie do najnowszego wpisu, a na mniejszym obszarze po prawej umieszczę pole logowania i spis odnośników do najnowszych artykułów na blogu. Oto kod HTML obszaru `feature_area`.

```
<section id="feature_area">
  <article id="blog_leadoff">
    <div class="inner">
      <h4>7 września 2012</h4>
      <a href="#"><h3>Klasy CSS w jQuery</h3></a>
      
      <p class="css_cols3">Sintus at neque in magna...</p>
    </div>
  </article>
  <aside>
    <form autocomplete="off" class="signin"
      action="process_form.php" method="post">
      <fieldset>
        <legend><span>Pobierz kody i aktualizacje</span>
        </legend>
        <section>
          <label for="email">E-mail</label>
          <input type="text" id="email" name="email" />
        </section>
        <section>
          <label for="password">Hasło</label>
```

wymagany znacznik form ————

kontener zbioru kontroltek ————

oznaczenie tekstowe zbioru kontroltek ————

kontener pomagający przy stylizacji kontrolki, oznaczenia i wskazówki —  
 atrybut for (o takiej samej wartości, jak identyfikator kontrolki) łączy —  
 oznaczenie z kontrolką

wartość atrybutu text sprawia, ————  
 że kontrolka wyświetlana jest jako pole tekstowe

wpisywane znaki wyświetlane są  
jako punktory

```
<input type="password" id="password"
name="password" maxlength="20" />
```

przycisk zatwierdzający

```
</section>
<section>
  <input type="submit" value="Zapisz się" />
  <p class="signup">Nie jesteś zarejestrowany?
  <a href="#">
    Zrób to teraz!</a></p>
</section>
```

```
</fieldset>
```

```
</form>
```

```
<nav>
```

```
<!-- odnośniki do wpisów -->
```

```
</nav>
```

```
</aside>
```

```
</section>
```

Element oznaczający tę sekcję jest kontenerem rozciągniętym na całą szerokość strony. Zawarte w nim elementy `article` i `aside` będą sływać na boki, zajmując miejsca obok siebie.

sprawia, że element obejmuje  
pływające elementy dzieci

```
section#feature_area {
  overflow:hidden;
  margin:16px 0 0;
  padding:0 0 10px;
}
```

odstęp między nagłówkiem  
a obszarem treści

```
section#feature_area article {float:left; width:66%;}
```

```
section#feature_area aside {float:right; width:34%;}
```

W ten sposób tworzę dwie kolumny w kontenerze. Zauważ, że określiłem ich szerokość wartościami procentowymi, gdyż chcę, by stronę można było oglądać na różnych urządzeniach, m.in. tabletach i smartfonach. Wróć do tego w następnym rozdziale. Utworzone kolumny mają zatem szerokość będącą określonym odsetkiem szerokości kontenera.

Wiem, że na tym etapie muszę osadzić `div` (który wyróżniłem w powyższym kodzie) w elemencie `article`, żeby utworzyć obramowanie wokół treści. Przyjrzyjmy się teraz stylom obszaru `article`.

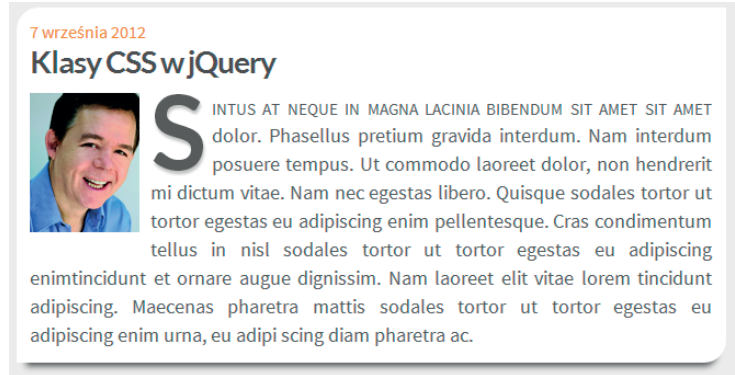
kontener z zaokrąglonymi rogami i cieniem	<pre>section#feature_area article .inner {     padding:12px;     background:#fff;     border-radius:20px 0;     box-shadow:0 12px 8px -9px #555; }</pre>
odnośnik z nagłówka	<pre>section#feature_area article a {text-decoration:none;}</pre>
zdjęcie	<pre>section#feature_area article img {     float:left;     padding:0 10px 10px 0; }</pre>
data	<pre>section#feature_area article h4 {     font-family:"Source Sans Pro", helvetica, sans-serif;     font-weight:400;     font-size:1em;     color:#f58c21;     letter-spacing:-.025em; }</pre>
nagłówek bloga	<pre>section#feature_area article h3 {     font-family:'Lato', helvetica, sans-serif;     font-weight:700;     font-size:1.75em;     color:#555;     margin:0 0 12px 0;</pre>

```

        letter-spacing: -.05em;
    }
tekst główny ————— | section#feature_area article#blog_leadoff p {
        font-family: "Source Sans Pro", helvetica, sans-serif;
        font-weight: 400;
        font-size: 1.1em;
        line-height: 1.5em;
        color: #616161;
        text-align: justify;
    }
inicjał ————— | section#feature_area article#blog_leadoff p::first-letter {
        font-family: 'Lato', helvetica, sans-serif;
        font-weight: 700;
        font-size: 4.5em;
        float: left;
        margin: .05em .05em 0 0;
        line-height: 0.6;
        text-shadow: 1px 3px 3px #ccc;
    }
cień wyświetlany jest w IE10
i nowszych wersjach przeglądarki — | section#feature_area article#blog_leadoff p::first-line {
        font-variant: small-caps;
        font-size: 1.2em;
    }
kapitałki w pierwszym wierszu — | section#feature_area article#blog_leadoff p::first-line {
        font-variant: small-caps;
        font-size: 1.2em;
    }
prawa kolumna ————— | section#feature_area aside {
        width: 34%;
        float: right;
    }

```

RYSUNEK 7.10. Wystylizowany element `article` w obszarze `feature_area`



Powyższe style odnoszą się do kilku elementów, które już wcześniej omówiłem. Warto zwrócić uwagę na odnośnik `a`, obejmujący nagłówek `h3`. Umieszczanie elementów blokowych w elementach liniowych było kiedyś absolutnie niedopuszczalne, ale w HTML5 odnośnikiem można objąć dowolny element, dzięki czemu oczywiście łatwiej określać klikalność elementów.

Jak widać na **rysunku 7.10**, obraz spływa w lewo i jest obłany tekstem. Użyłem zestawienia inicjału z kapitalikami, które zaprezentowałem w rozdziale 4., aby urozmaicić oprawę i gładko przejść od nagłówka do tekstu. Inicjałowi nadałem także cień, aby wybijał się nieco ze strony. Zajmę się teraz stylizacją formularza.

## Stylizacja pola logowania

Czytelników, którzy chcą pobrać kody z moich książek, proszę o rejestrację, abym mógł im wysyłać aktualności i utrzymać z nimi kontakt. Na stronie głównej znajduje się pole logowania z odnośnikiem do formularza rejestracyjnego dla nowych użytkowników. Formularz na stronie głównej składa się z takiego samego kodu, jak formularz omówiony w rozdziale 6. Oto jego kod HTML:

```
<form autocomplete="off" class="signin" action="process_
form.php" method="post">
  <fieldset>
    <legend><span>Zapisz się, by pobrać kody i otrzymywać
aktualizacje</span></legend>
    <section>
      <label for="email">E-mail</label>
```

oznaczenie tekstowe zbioru kontroltek

pole na adres e-mailowy

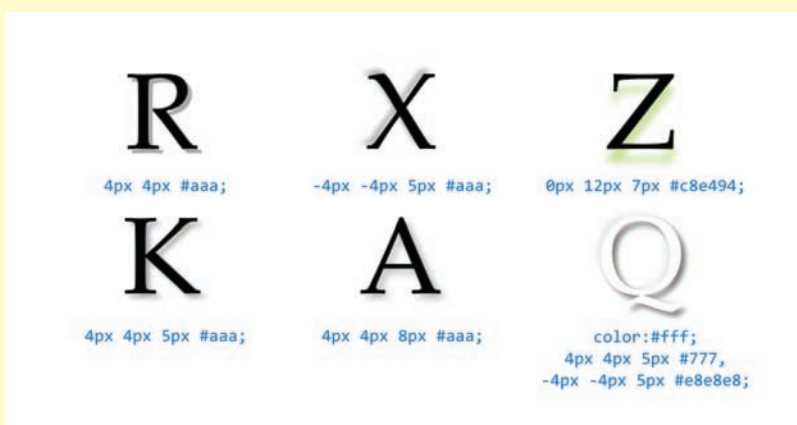
atrybut `for` (o takiej samej wartości, jak identyfikator kontrolki) łączy oznaczenie z kontrolką

## Cienie tekstu

Cienie tekstu są bardzo podobne do cieni pól, które omówiłem wcześniej w ramce „Cienie pól”. Oto składnia kodu, którym się je tworzy:

```
text-shadow:4px 4px 5px #aaa;
```

Kolejność stylów jest następująca: przesunięcie w poziomie, przesunięcie w pionie, rozmycie i kolor. W odróżnieniu od cieni pól, cienie tekstu nie mają ustawienia rozszerzenia. Musisz podać przynajmniej przesunięcie w poziomie i pionie oraz kolor, co tworzy cień o wyrazistej krawędzi, określonej szerokości i kolorze. Określając ujemne wartości przesunięcia w prawo i w dół, przesuwasz cień w lewo i w górę. Można podać więcej niż jedną deklarację cienia, oddzielając je od siebie przecinkami. Na **rysunku 7.11** widnieją różne przykłady tego, co można tą właściwością uzyskać. O bardziej zaawansowanym zastosowaniu cieni tekstu przeczytasz w moim e-booku *Visual Stylin' with CSS3*.



RYSUNEK 7.11. Przy użyciu różnych wartości, zarówno dodatnich, jak i ujemnych, można uzyskać różne efekty cienia tekstu

wartość atrybutu text sprawia, że kontrolka wyświetlana jest jako pole tekstowe

pole hasła

tekst ukryty, kiedy element nie przynależy do klasy error

przycisk zatwierdzający

```
<input type="text" id="email" name="email" />
```

```
</section>
```

```
<section>
```

```
<label for="password">Hasło</label>
```

```
<input type="password" id="password" name="password" maxlength="20" />
```

```
<p class="direction">Błędna nazwa użytkownika lub hasło</p>
```

```
</section>
```

```
<section>
```

```

        <input type="submit" value="Zapisz się" />
        <p class="signup">Nie jesteś zarejestrowany?
        <a href="#">
        Zrób to teraz!</a></p>
    </section>
</fieldset>
</form>

```

Z przykładu formularza w rozdziale 6. pobrałem jedynie nieodzwonione fragmenty kodu CSS, które dostosowałem do potrzeb tego formularza.

```

ogólna szerokość formularza — form.signin {
    width:19em;
    float:right;
    background:#fff;
    border-radius:10px 0 10px 0;
    box-shadow:0 12px 8px -9px #555;
}

usuwa domyślne obramowanie — .signin fieldset {border:0; margin:10px 14px;}
elementu fieldset

.singin legend span {
    font-family:'Lato', helvetica, sans-serif;
    font-weight:700;
    font-size:1.3em; line-height:1.1em;
    color:#4eb8ea;
    letter-spacing:-.05em;
}

.singin section {
    overflow:hidden;
    padding:.25em 0;
}

.singin section label {
    font-family:"Source Sans Pro", helvetica, sans-serif;

```

sprawia, że element obejmuje kontrolkę i oznaczenie formularza —

odstęp między elementami formularza —



szerość kolumny oznaczenia	—	<code>font-weight:400;</code>
		<code>float:left;</code>
prawy margines oddala tekst od kontrolki	—	<code>width:5em;</code>
		<code>margin:.5em .3em 0 0;</code>
		<code>font-size:1em; line-height:1.1;</code>
		<code>color:#555;</code>
		<code>}</code>
		<code>.signin section input {</code>
		<code>float:right;</code>
szerość kolumny kontrolki	—	<code>width:10.5em;</code>
		<code>margin:.2em 0 0 .5em;</code>
tworzy przestrzeń wokół tekstu kontrolki	—	<code>padding:3px 10px 2px;</code>
		<code>color:#555;</code>
		<code>font-size:.8em;</code>
usuwa niebieską obwódkę, która domyślnie pojawia się po sfokusowaniu elementu	—	<code>outline:none;</code>
		<code>border-radius:10px 0 10px 0;</code>
		<code>}</code>
usuwa żółte tło w przeglądarkach opartych na Webkit	—	<code>input:-webkit-autofill {color:#fff !important;}</code>
		<code>.signin section input[type=submit] {</code>
wyrównuje przycisk względem prawych krawędzi kontrolki	—	<code>float:right;</code>
usuwa ustawienia szerokości odziedziczone po innych polach	—	<code>width:auto;</code>
		<code>margin:0 2px 3px 0;</code>
		<code>padding:0px 8px 3px;</code>
		<code>font-size:1em;</code>
		<code>font-weight:800;</code>
		<code>color:#fff;</code>
		<code>border:none;</code>
		<code>background-color:#d6e636;</code>

```

        box-shadow:1px 1px 2px #888;
    }

```

tekst „Nie jesteś zarejestrowany?” → `.signin section p {`

```

    float:right;
    clear:both;
    margin:.2em 0 0;
    text-align:right;
    font-size:.8em;
    line-height:1;
    color:#555;
}

```

odnośnik do formularza rejestracyjnego → `.signin section p a {color:#333;}`

```

.signin section p a:hover {
    color:#777;
    text-decoration:none;
}

```

komunikat o błędzie → `.signin section p.direction.error {`

```

    display:block;
}

```

podświetla tekst wskazówki na czerwono, gdy dodana jest klasa error → `color:#f00;`

```

}

```

ukrywa komunikat o błędzie → `.signin section p.direction {display:none;}`

RYSUNEK 7.12. Wystylizowany formularz logowania z komunikatem o błędzie

The image shows a login form with the following elements:

- Header: **Pobierz kody i aktualizacje**
- Input fields: "E-mail" and "Hasło" (Password).
- Error message: **Błędna nazwa użytkownika lub hasło** (Incorrect username or password).
- Submit button: **Zapisz się** (Sign up).
- Footer: **Nie jesteś zarejestrowany? [Zrób to teraz!](#)**

Niezależnie od stopnia złożoności formularza, utworzenie go zawsze wymaga wielu linijek kodu. Kod w tym przykładzie w większości jest jednak dość prosty, a komentarze objaśniają jego najważniejsze fragmenty. Chciałbym jeszcze tylko zwrócić uwagę na komunikat o błędzie, który pozostaje ukryty, dopóki nie jest potrzebny (rysunek 7.12). Wystarczy dodać klasę `error` do elementu `p` (który już jest przypisany do klasy `direction`), aby komunikat się pojawił. Tym niemniej założenie jest takie, że ta klasa ma być w razie potrzeby dodawana automatycznie przez skrypt służący do walidacji danych formularza. Jako osoba odpowiedzialna za stworzenie interfejsu użytkownika, odpowiadasz za umieszczenie na stronie ukrytego zwykle elementu HTML z informacją o błędzie i opracowanie kodu CSS służącego do jego wyświetlenia — zadaniem programistów jest rozpracowanie, jak i kiedy nadać klasę przywołującą kod CSS odpowiedzialny za wyświetlenie komunikatu.

## Odnośniki do wpisów

Pod formularzem znajdują się odnośniki do wpisów. Jak zwykle zamieściłem je w nieuporządkowanej liście.

```
<nav>
  <h3>Ostatnie wpisy</h3>
  <ul>
    <li><a href="#">Z-index &mdash; problemy się
      nawarstwiają</a></li>
    <li><a href="#">Jak używać box-image</a></li>
    <li><a href="#">Cienie w CSS3</a></li>
  </ul>
</nav>
```

A oto CSS:

```
section#feature_area nav {
  ogólna szerokość kontenera — width:19em;
  odstępy pod i nad elementem — float:right;
  dopełnienie pod i nad elementem — margin:15px 0 0;
  wyrównuje element do prawej — padding:.6em 0em .75em;
  krawędzi elementu section — background:#fff;
  border-radius:10px 0 10px 0;
```

```

        box-shadow:0 12px 8px -9px #555;
    }
#feature_area nav h3 {
pozioma przestrzeń na tytuł — padding:0 14px;
                                font-family:'Lato', helvetica, sans-serif;
                                font-weight:700;
                                font-size:1.3em;
                                text-align:left;
                                color:#aaa;
                                letter-spacing:-.05em;
}

#feature_area nav ul {margin:0 0 0 20px;}
#feature_area nav li {
kontekst pozycjonowania — position:relative;
punktatorów
                                list-style-type:none;
}

#feature_area nav li::before {
autorskie punktory — content: "";
pusty ciąg tekstowy jako treść — position:absolute;
                                height:10px; width:10px;
pozycjonowanie względem — left:12px; top:12px;
elementów listy
wielkość punktora — border-radius:5px 0 5px 0;
położenie punktora — background-color:#d6e636;
                                box-shadow:1px 1px 2px #888;
}

#feature_area nav li a {
odnośnik zajmuje całą szerokość — display:block;
kontenera

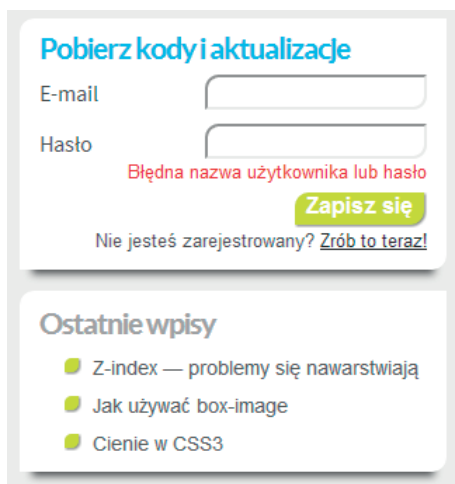
```

usuwa domyślne podkreślenie — `text-decoration:none;`  
`font-size:.9em;`  
`color:#616161;`  
`}`

`#feature_area nav li a:hover {color:#000;}`

Wystylizowany obszar odnośników znajduje się bezpośrednio pod obszarem pola logowania. Obydwa obszary zawarte są w elemencie `aside`, który pływa obok elementu `article` (rysunek 7.13).

RYSUNEK 7.13. Ukończony element `aside` z odnośnikami do wpisów



Sprawiając, by element `nav` spływał tak samo jak znajdujący się nad nim element `form`, zyskuję możliwość umieszczenia go bezpośrednio pod formularzem. O ile element ten jest wystylizowany jak standardowa lista, to jego punktory są ciekawe. Chciałem, żeby odzwierciedlały motyw „dwóch zaokrąglonych, dwóch zwyczajnych” rogów, którym charakteryzuje się oprawa graficzna strony. Zamiast więc tworzyć osobny obraz punktora, użyłem pseudoelementu `::before` do utworzenia 10-pikselowego, kwadratowego elementu i zaokrąglenia jego rogów. Drobnny, jednopikselowy cień wybija nieco punktory ze strony.

## Obszar książek

Okładki czterech książek wyświetlone są w rzędzie u dołu strony. Na tym obszarze mamy do czynienia z kilkoma interesującymi elementami, takimi jak chmurki i obrócony tekst. Oto kod HTML:

```

<section id="book_area">
<!-- poniższy element powtórzony jest jeszcze trzykrotnie -->
<article class="left">
  <div class="inner">
    obrócony tekst ————— | <h3>HTML5 + CSS3</h3>
                                | 
    wskazówka ————— | <aside>
                                |
                                | <ol>
                                |   <li><a href="#">Pobierz kody</a></li>
                                |   <li><a href="#">Spis treści</a></li>
                                |   <li><a href="#">Kup książkę</a></li>
                                | </ol>
                                | </aside>
                                | </div>
  </article>
<!-- koniec powtórzeń kodu ramek z książkami -->
</section>

```

Powyższy fragment kodu jest bardzo prosty. Jak widać w komentarzu, powtórzony jest on jeszcze trzy razy, odpowiednio dla kolejnych książek. Przyjrzyjmy się teraz powiązanemu kodowi CSS. Zaczynam od utworzenia układu obszaru z książkami i obróconego tekstu, a następnie zajmuję się chmurkami.

```

element o pełnej szerokości ————— | section#book_area {
                                        |   clear:both;
                                        |   border-radius:20px 0px 20px 0px;
                                        |   border:1px solid #f58c21;
odstęp nad i pod elementem ————— |   margin:8px 0 16px;
                                        |   overflow:hidden;
                                        | }

```

cztery kolumny na książkę	→	<code>#book_area article {</code> <code>float:left;</code> <code>width:25%;</code> <code>padding:10px 0;</code> <code>background:none;</code> <code>}</code>
kontener obejmujący książki	→	<code>#book_area article .inner {</code>
kontekst pozycjonowania chmurki	→	<code>position:relative;</code>
obejmuje każdą książkę	→	<code>width:140px;</code>
wyśrodkowuje każdą książkę w jej elemencie article	→	<code>margin:0 auto;</code> <code>}</code>
obrócony tekst	→	<code>#book_area .inner h3 {</code> <code>position:absolute;</code> <code>width:160px;</code> <code>left:112%; bottom:5px;</code> <code>transform:rotate(-90deg);</code> <code>transform-origin:left bottom;</code> <code>color:#ccc;</code> <code>font-family:'Lato', helvetica, sans-serif;</code> <code>font-weight:900;</code> <code>font-size:1.4em;</code> <code>text-align:left;</code> <code>}</code>
umieszcza tekst po prawej stronie książki	→	
obraca tekst (konieczne jest dodanie prefiksów)	→	
wskazuje oś obrotu na początku tekstu (konieczne jest dodanie prefiksów)	→	
inne przesunięcie dla węższej okładki	→	<code>#book_area article.right:last-child h3 {left:85%;}</code>
cień pod każdą książką	→	<code>#book_area article img {box-shadow:0 12px 8px -9px #555;}</code>

W elemencie `section` o pełnej szerokości (`#book_area`) zamieściłem cztery pływające elementy `article`, każdy o szerokości równej 25%. W każdym z nich wyśrodkowałem ustalonej szerokości `div` klasy `inner`, zawierający obraz książki. Uzyskałem w ten sposób rząd okładek z ładnymi odstępami (rysunek 7.14). Element `aside`, który przekształcę w chmurkę, jest obecnie ukryty.



RYSUNEK 7.14. Każda książka w tym obszarze oznaczona jest skrótowym opisem, obróconym do pionu i zamieszczonym przy jej prawej krawędzi

Położenie obróconego tekstu określone jest dwiema funkcjami właściwości CSS3 `transform`. Pierwsza z nich, `transform-origin`, wskazuje lewy dolny róg pola elementu `h3` jako punkt początkowy przejścia. Punkt początkowy działa tak, jakbyś w jego miejscu wbił w element szpilkę. Obróciłem następnie element `h3` o dziewięćdziesiąt stopni przy użyciu funkcji `rotate` właściwości `transform`, a na koniec przesunąłem go o pięć pikseli w górę, by dostosować jego położenie. Krótkie omówienie właściwości `transform` znajdziesz w ramce „Przekształcenia w CSS3”, a więcej na ten temat przeczytasz w moim e-booku *Visual Stylin' with CSS*.

Nadszedł czas, by dodać chmurki do książek. Chmurkę przedstawiłem w rozdziale 6. nieco rozwinąłem na dwa sposoby. Po pierwsze, chmurki książek znajdujących się w prawej połowie strony będą wyświetlane po lewej stronie okładek, aby nie były ucinane przez okno przeglądarki. Po drugie, strzałkę przylegającą do chmurki wystylizowałem tak, by uzyskać wrażenie, że ona także jest objęta obramowaniem pola, a tym samym jest jego częścią. Jak to często bywa z takimi pozornie drobnymi, lecz istotnymi szczegółami, uzyskanie ich wymaga dużej ilości kodu.

W podanym powyżej kodzie widać, że każdemu elementowi `article` z obrazami książek nadałem klasy `left` i `right`, aby móc im przypisać kod CSS odnoszący się do rozmieszczenia chmurki oraz ich strzałek po lewej lub prawej stronie. A oto kod!

```
tutaj zaczynają się style
wspólne dla wszystkich chmurki — #book_area article aside {
ukrywa chmurki — display:none;
pozycjonowanie względem — position:absolute;
elementu div klasy inner, — z-index:2;
który otacza obraz
szerokość chmurki — width:200px;
```



## Przekształcenia w CSS3

Jeżeli pracowałeś z edytorami graficznymi takimi jak Adobe Illustrator lub Adobe Fireworks, to prawdopodobnie już obracałeś, skalowałeś i pochylałeś tekst lub innego rodzaju elementy. Teraz możesz uzyskać te same efekty w przeglądarce, używając przekształceń CSS3 (które widać na **rysunku 7.15**).

Do tworzenia przekształceń w CSS3 służą dwie właściwości: `transform` i `transform-origin`. Zacznijmy od omówienia właściwości `transform`.

Właściwość `transform` opiera się w działaniu nie na zwykłych wartościach, ale na funkcjach. Funkcje pozwalają Ci na podanie rodzaju przekształcenia oraz wartości, z jakich ma być obliczone.

Format przekształcenia wygląda następująco: `transform: nazwaFunkcji(wartość liczbowa lub x, y)`.

Oto funkcje przekształceń:

- `scale` — powiększa lub zmniejsza element. Wartości większe niż 1 powiększają element, a mniejsze zmniejszają go. Przykład: `transform:scale(1.5)`.
- `rotate` — obraca element o podaną liczbę stopni. Wartości dodatnie obracają go zgodnie z ruchem wskazówek zegara, a ujemne w przeciwną stronę. Przykład: `transform:rotate(-30deg)`.
- `skew` — przechyla element względem jego osi x lub y. Kiedy podana jest jedna wartość, przechylenie względem osi y nie zachodzi. Przykład: `transform:skel(5deg, 50deg)`.
- `translate` — przesuwa obiekt o podaną odległość po jego osi x lub y. Przypomina to pozycjonowanie względne, ponieważ miejsce pierwotnie zajęte przez element nie znika. Przykład: `transform:translate(-50px, 20px)`.

Właściwość `transform-origin` określa punkt, względem którego element ma zostać przekształcony. Domyślnie jest to punkt pośrodku elementu w poziomie i pionie, więc jeśli obrócisz element, to będzie się on zachowywał, jakby wbito w jego środek szpilkę, i obracał się wokół niej. Właściwością `transform-origin` możesz wybrać inny punkt w elemencie przy użyciu słów kluczowych (`top`, `right` itd.) albo podać dodatnie lub ujemne wartości liczbowe. W ten sposób punkt przekształcenia można umieścić nawet poza polem elementu.



RYSUNEK 7.15. Oto kilka przykładów przekształceń

przeźren wokół treści chmurki	<pre>background:#fff; padding:10px 2px 5px; border:2px solid #f58c21; border-radius:10px 0px 10px 0px; box-shadow:4px 4px 16px #555; color:#555; font-family:"Source Sans Pro", helvetica, sans-serif; font-size:.8em; line-height:1.5em; }</pre>
wyświetla chmurkę, kiedy kursor znajduje się nad książką	<pre>#book_area article:hover aside {display:block;} #book_area article aside li {</pre>
<p> pionowe odstępy między elementami listy i dopełnienie z lewej</p> <p> usuwa domyślne punktory elementów listy</p>	<pre>padding:.25em 0 .75em 1em; list-style-type:none; line-height:1.2em; }</pre>
odnośniki tekstowe	<pre>#book_area article aside li a { text-decoration:none; font-size:1.2em; color:#616161; }</pre>
podświetla odnośniki, nad którymi znajduje się kursor	<pre>#book_area article aside li a:hover { color:#333;</pre>
koniec stylów wspólnych dla wszystkich chmurzek	<pre>}</pre>
<p> dwie chmurki książek znajdujących się po lewej</p> <p> umieszcza chmurkę po prawej stronie obrazów znajdujących po lewej</p>	<pre>#book_area article.left aside { left:84%; top:14px; }</pre>

dwie chmurki książek znajdujących się po prawej —> `#book_area article.right aside {`  
 umieszcza chmurkę po lewej stronie obrazów znajdujących się po prawej —> `right:84%;`  
`top:14px;`  
`}`

pole pomarańczowego trójkąta —> `#book_area article aside::after {`  
 pusty ciąg tekstowy — wymagane jest podanie jakiegokolwiek treści —> `content:"";`  
 położenie określone względem chmurki —> `position:absolute;`  
`top:33px;`  
`border:12px solid;`  
 zmniejsza pole, by stworzyć trójkąt —> `height:0px; width:0px;`  
`}`

określa położenie i kolor trójkątów chmurek książek, które znajdują się po lewej —> `#book_area article.left aside::after {`  
`right:100%;`  
`border-color:transparent #f58c21 transparent transparent;`  
`}`

określa położenie i kolor trójkątów chmurek książek, które znajdują się po prawej —> `#book_area article.right aside::after {`  
`left:100%;`  
`border-color:transparent transparent transparent #f58c21;`  
`}`

pole białego trójkąta —> `#book_area article aside::before {`  
 pusty ciąg tekstowy — wymagane jest podanie jakiegokolwiek treści —> `content:"";`  
 położenie określone względem chmurki —> `position:absolute;`  
`top:37px;`  
`border:8px solid;`  
 zmniejsza pole, by stworzyć trójkąt —> `height:0px; width:0px;`  
 sprawia, że biały trójkąt znajduje się na szczycie stosu —> `z-index:100;`  
`}`

styl, położenie i kolor białego trójkąta chmurki książek, które znajdują się po lewej

```
#book_area article.left aside::before {
    right:100%;
    border-color:transparent white transparent transparent;
}
```

styl, położenie i kolor białego trójkąta chmurki książek, które znajdują się po prawej

```
#book_area article.right aside::before {
    left:100%;
    border-color:transparent transparent transparent white;
}
```


Chmurka jest elementem pozycjonowanym bezwzględnie w stosunku do elementu `div` klasy `inner`, który zawiera książkę. W przykładzie w rozdziale 6. z boku chmurki zwyczajnie umieściłem czerwony trójkąt (**rysunek 6.24**). W tym przykładzie znacznie ulepszyłem ten efekt: trójkąt wygląda, jakby wybijał się z chmurki i był objęty jej obramowaniem (**rysunek 7.16**). Uzyskałem ten rezultat, tworząc nieco mniejszy, biały (czyli takiego samego koloru, jak tło chmurki) trójkąt nad pomarańczowym trójkątem i wyrównując je w pionie. Pomarańczowy trójkąt służy zatem do uzyskania efektu obramowania.

Biały trójkąt utworzyłem pseudoelementem `::before`, a pomarańczowy pseudoelementem `::after`. Do precyzyjnego wyrównania ich położenia użyłem pozycjonowania bezwzględnego oraz właściwości `z-index`. Rezultat jest bardzo estetyczny i łatwo uwierzyć, że chmurka i trójkąt stanowią razem jeden element.

Trudne może tu jednak być rozeznanie się w tym, co właściwie znajduje się z lewej, a co z prawej. Chmurki dwóch książek po lewej są na przykład zamieszczone po prawej stronie obrazów, a ich trójkąty z kolei znajdują się po ich lewej stronie. W przypadku dwóch książek z prawej położenie elementów jest odwrócone.



RYSUNEK 7.16. W chmurkach przylegających do każdej książki znajdują się odnośniki do dodatkowych informacji; zastosowanie chmurkek pozwala na zachowanie miejsca

 Omówione tu podejście oparte jest na maksymie DRY (ang. don't repeat yourself, czyli „nie powtarzaj się”), a jego celem jest stworzenie „jednego, nadrzędnego źródła” definiującego wszystkie fragmenty danych.

Pracę od ogółu do szczegółu, o której wcześniej wspomniałem, widać tutaj w działaniu. Jak widać w komentarzach do powyższego kodu, najpierw podałem kod CSS odnoszący się do wszystkich chmurkek, czyli style określające ich wielkość, dopełnienia, kolor obramowań i treść. Dalej podałem kod służący do rozmieszczenia dwóch chmurkek po prawej stronie pierwszej i drugiej książki oraz dwóch chmurkek po lewej stronie trzeciej i czwartej książki. Dalej znajdują się style tworzące trójkąty, które występują przy każdej chmurce. Na końcu podaję style trójkątów chmurkek dwóch książek po lewej oraz dwóch książek po prawej.

Choć trzeba było dużo wytłumaczyć, powyższy kod jest uporządkowany logicznie i nie występuje w nim zjawisko powtarzania się całych zbiorów stylów dla każdej chmurki. Warto poświęcić nieco czasu na przestudiowanie lub zrekonstruowanie tego przykładu, aby zyskać lepsze zrozumienie tej struktury. Ułatwi Ci to kodowanie wielu elementów o podobnych, ale nieco odmiennych stylach i zachowaniach. Przede wszystkim, takie podejście pozwala na uzyskanie kodu zrozumiałego i łatwego w obsłudze dla Ciebie i innych. Zakończmy tworzenie strony, dodając stopkę.

## Stopka

Stopka strony jest dobrym miejscem na informację o autorze i odnośniki do materiałów związanych ze sprawami formalnymi, takich jak zrzeczenie się odpowiedzialności, regulamin, informacje kontaktowe, polityka prywatności i informacje o prawach autorskich. Oto kod HTML:

```

<footer>
  <p>Szablon CSS z książki <a href="http://www.
  stylinwithcss.com">
  <em>CSS. Witryny internetowe szyte na miarę, wydanie
  trzecie</em>
</a> Charlesa Wyke-Smitha</p>
<nav>
  <ul>
    <li><a href="#">Polityka prywatności</a></li>
    <li><a href="#">Kontakt</a></li>
  </ul>
</nav>
</footer>

```

Oto kod CSS:

```

footer {
  odstępy pod i nad elementem — padding: .5em 0 .35em 0;
  wyśrodkowuje treść — text-align:center;
  border-radius:10px 0px 10px 0px;
  background:#fff;
  box-shadow:0 12px 8px -9px #555;
}

stylizacja tekstu — footer p {
  font-family:"Source Sans Pro", helvetica, sans-serif;
  font-weight:400;
  font-size:.85em;
  letter-spacing:-.05em;
  color:#555;
}

odnośnik w tekście — footer p a {
  font-family:"Source Sans Pro", helvetica, sans-serif;
  font-style:italic;
  font-weight:700;
  font-size:1em;
  color:#4eb8ea;
}

```

		<code>text-decoration:none;</code>
		<code>}</code>
		<code>footer p a:hover {</code>
		<code>color:#777;</code>
		<code>}</code>
lista odnośników	→	<code>footer ul {</code>
sprawia, że pole elementu ściśle obejmuje listę	→	<code>display:inline-block;</code>
		<code>margin:4px 0 0;</code>
		<code>}</code>
		<code>footer li {</code>
usuwa domyślne punktory	→	<code>list-style-type:none;</code>
rozmieszcza elementy listy poziomo	→	<code>float:left;</code>
		<code>font-family:"Source Sans Pro", helvetica, sans-serif;</code>
		<code>font-weight:400;</code>
		<code>font-size:.85em;</code>
		<code>}</code>
		<code>footer li + li a {</code>
linie oddzielające odnośniki	→	<code>border-left:1px solid #ccc;</code>
		<code>}</code>
		<code>footer li a {</code>
usuwa domyślne podkreślenie odnośników	→	<code>text-decoration:none;</code>
		<code>color:#aaa;</code>
odstępy między odnośnikami	→	<code>padding:0 5px;</code>
		<code>}</code>
		<code>footer a:hover {</code>
		<code>color:#777;</code>
		<code>}</code>

RYSUNEK 7.17. W wystylizowanej stopce znajduje się element tekstowy i lista

Nie ma tu niczego, czego byś wcześniej nie widział. Treść została wyśrodkowana właściwością `text-align:center` (rysunek 7.17). Tę właściwość dziedziczy akapit, którego tekst zostaje w nim wyśrodkowany. Zauważ, że właściwość `text-align:center` standardowo nie wyśrodkowywałaby listy odnośników w ten sposób, ponieważ elementy blokowe standardowo zajmują całą szerokość kontenera. Element `ul` jednak ściśle obejmuje elementy `li`, ponieważ nadałem mu właściwość `display:block-inline`. W ten sposób zyskuje on określoną szerokość i jest wyśrodkowywany ustawieniem `text-align:center`. Jak być może pamiętasz z kodu stylizującego menu, wykorzystanie właściwości `display:block-inline` sprawia, że szerokość pozostaje płynna, dzięki czemu lista będzie wyśrodkowywana także po dodaniu do niej lub odjęciu od niej elementów. Zauważ, że zastosowanie marginesów o wartości `auto` zamiast właściwości `text-align:center` wyśrodkowałoby listę w równie dobrym stopniu.

## Podsumowanie

Niniejszym kończymy rozdział, a tym samym tworzenie layoutu. Mam nadzieję, że przekonałeś się, że rozpoczęcie pracy od utworzenia struktury przy pomocy technik zaprezentowanych w rozdziale 5., a następnie dodawanie do strony komponentów opisanych w rozdziale 6. pozwala na szybkie tworzenie pełnoprawnych witryn. Nad przykładową stroną będę pracował dalej w kolejnym rozdziale, ponieważ współcześnie witryny internetowe nie są wyświetlane jedynie w formie szerokich layoutów pasujących do dużych monitorów takich jak ten, na którym pracuję. Twoje strony muszą być skalowalne i reagować na wielkość ekranów urządzeń, na jakich się je ogląda. Utworzenie kilku layoutów o różnych wymiarach i staranie się, by właściwy layout trafiał na właściwy ekran, nie jest dobrym pomysłem. Zamiast tego lepiej umieć tworzyć strony, które reagują na warunki zewnętrzne i wczytują odpowiedni kod CSS do stylizacji kodu HTML w taki sposób, by layout pasował do urządzenia, na którym się go wyświetla. W ostatnim rozdziale tej książki zademonstruję tworzenie skalowalnych witryn.



# Skorowidz

960 Grid, 152

## A

Adobe Dreamweaver, 7, 18  
 Adobe Fireworks, 264  
 Adobe Illustrator, 264  
 adres URL, 9, 44  
 akapit, 2, 5, 10, 19, 20, 71  
 Apple Retina, 275  
 arkusz stylów, 25, 50, 274, 291, 293
 

- autorski, 51
- przeglądarki, 50
- użytkownika, 50

## B

biblioteka fontów, 127, 129  
 brat, 32, 33, 46

## C

chmurka, 224, 267, 268  
 Chrome, 128, 223  
 cień, 241, 254  
 Clarke Andy, 277  
 cudzysłów, 19  
 cytat, 18

## D

debugowanie, 293  
 deklaracja, 26, 55
 

- waga, 52

 Document Object Model, *Patrz:* DOM  
 dokument HTML
 

- dołączanie stylów, 25
- struktura, 7, 20, 21

DOM, 20, 21, 22, 29  
 dopełnienie, 63, 66, 68, 71, 161, 163  
 dziecko, 6, 13, 21, 32, 33, 81  
 dziedziczenie, 49, 110, 119

## E

ekran, 273, 275, 277, 279, 282
 

- dotykowy, 287, 289

 element, 55, 50, *Patrz też:* znacznik
 

- blockquote, 18, 19, 146
- blokowy, 10, 15, 19, 86, 93, 125
  - szerokość, 15, 70, 153
- body, 16
- div, *Patrz:* znacznik div
- dopełnienie, *Patrz:* dopełnienie
- fieldset, 206, 210
- figcaption, 225
- figure, 225
- form, 205, 221
- generowany dynamicznie, 224
- liniowy, 10, 13, 16, 93, 125
- margines, *Patrz:* margines
- oblewianie tekstem, 76
- obramowanie, *Patrz:* obramowanie
- oczyszczający, 75
- pływający, 75, 76, 77, 78, 80, 194
- pozycjonowanie, 62, 87, 90
  - bezwzględne, 88, 91, 234, 267
  - stałe, 89
  - statyczne, 86
  - względne, 87
- przerośnięty, 168
- span, 147
- szerokość, 70, 73, 77, 244
- tabelowy, 10
- table, 178
- tekstowy, 2, 5
  - label, 207

## element

- tr, 178
- warstwa, *Patrz:* warstwa
- wyśrodkowanie, 244, 248
- z cieniem, *Patrz:* cień
- z zaokrąglonymi rogami, *Patrz:* zaokrąglone rogi
- zagnieżdżony, 16, 18
- złożony, 5

## encja, 19, 143

- &amp, 143
- &bdquo, 19
- &gt, 143
- &rdquo, 19

**F**

## Firebug, 294

## Firefox, 128

- Web Developer, 14, 16

font, 108, 110, 111, 112, *Patrz też:* tekst

- bezseryfowy, 110
- Embedded OpenType, 129
- internetowy, 126, 128, 129
- na serwerze, 108, 128
- na zewnętrznych serwisach, 108
- OpenType, 128
- Scalable Vector Graphics, 129
- seryfowy, 110
- TrueType, 128
- zainstalowane w systemie użytkownika, 108, 110

## Font Squirrel, 129, 130

## Fontspring, 130

## formatowanie graficzne, 62

## formularz, 5, 45, 201

- kodowanie, 209
- kontrolka, 207, 208
- oznaczenie, 207

## stylizacja, 210, 220

- wysyłanie, 206
- wyszukiwania, 221, 239

**G**

## Google Web Fonts, 127, 145

## gradient, 104

**I**

## identyfikator, 35, 38, 39

- związany z JavaScript, 39

## inicjał, 47, 147

## interfejs użytkownika, 186

## interlinia, 123

## Internet Explorer, 129, 130, 170, 171, 177, 295, 296, 297

## iPad, 273, 289

## iPhone, 273, 275, 285, 287, 289

## Irish Paul, 171, 297

**K**

## kanał alfa, 61

## kaskadowość, 50, 293

- zasady, 52, 54

## klasa, 35, 36, 39, 40, 41, 53, 182

- error, 215

## kod alternatywny, 295

## kod prezentacyjny, 164

## kolor, 27, 55

- HSL, 59, 61

- jasność, 60

- krycie, 61

- nasycenie, 60

- nazwa, 57

- tła, 95

## wartość

- numeryczna, 58
- procentowa, 59
- szesnastkowa, 58

koło barw, 60  
 komentarz warunkowy, 296  
 kompatybilność wsteczna, 7  
 kontener, 77, 78, 80, 81, 163, 233, 274  
 kratka, 208, 210

**L**

layout, 151, 152, 231  
   płynny, 172, 176, 274, 279  
   skalowanie, 272, 274, 278, 282, 287  
   szerokość, 152, 153  
   wielokolumnowy, 151, 152, 153, 154, 179  
     elastyczny, 151, 152  
     o stałej szerokości, 151, 152  
     płynny, 151, 152  
   wielorzędowy, 179, 180  
   wysokość, 152, 153  
 leading, 123  
 lista, 5, 187, 188

**M**

margines, 63, 67, 68, 71, 161, 245  
   jednostki miary, 69  
   scalenie, 68  
 media query, *Patrz:* zapytanie medialne  
 medium, 276, *Patrz też:* zapytanie medialne  
 menu, 186, 242  
   poziom  
     kolejny, 197  
     najwyższy, 192  
   poziome, 189  
   rozwijane, 191, 195, 196, 246  
   ścieżka wyboru, 200  
   wyboru, 208  
 model  
   formatowania graficznego, *Patrz:* formatowanie graficzne  
   obiektowy dokumentu, *Patrz:* DOM

  polowy, 62, 70  
   RGB, *Patrz:* RGB  
 Modernizr, 101, 248, 288, 297

**N**

nagłówek, 2, 5, 10, 12  
   h1, 2, 5, 21, 237

**O**

obramowanie, 63, 65, 71, 161, 163  
 obraz, 9, 10, 13  
   elastyczny, 274  
   tła, 95, 96, 102  
 odniesienie, 3  
 odnośnik, 9, 10, 38, 44, 186  
   do wpisów, 258

**P**

pole, *Patrz:* element logowania, 253  
 potomek, 49  
 pozycjonowanie, *Patrz:* element pozycjonowanie  
 prefiks, 103  
 projektowanie skalowalne, 272, 274, 278, 282, 289  
 przejście CSS, 222  
   dodawanie, 223  
 przełącznik, 208, 210  
 przezroczystość, 61, 229, 247  
 przodek, 32, 49  
 pseudoelement, 44, 47, 147  
 pseudoklasa, 43  
   focus, 45  
   interfejsu, 43  
   precyzja, 44, 53  
   strukturalna, 43, 46  
   target, 45

**R**

- reguła CSS, 24, 26, 27, 55
  - !important, 52
  - @font-face, 128, 130
  - @import, 25
  - @media, 274, 275, 277
- deklaracja, *Patrz:* deklaracja
- selektor, *Patrz:* selektor
- rem, 114
- Responsive Web Design, *Patrz:* projektowanie skalowalne
- RGB, 27, 55, 61
- rodzic, 6, 13, 46, 78, 79, 80
- RWD, *Patrz:* projektowanie skalowalne

**S**

- Safari, 128, 223
  - Mobile, 287
- selektor, 26, 35, 44, 53, 147
  - atrybutu, 28, 41, 42
    - nazwy, 41
  - gwiazdkowy, 166
  - identyfikatora, 28, 41
  - klasy, 28, 36, 41, 53
  - kontekstowy, 28, 29, 30, 32, 35, 44, 297
  - niepierworodny, 188
  - potomka, *Patrz:* selektor kontekstowy
  - precyzja, 53
  - pseudoklasy, 44
  - uniwersalny, 34
  - wieloklasowy, 37
- Shapira Isaac, 289
- smartfon, 282
- stopka, 268
- stos, 227
- strona HTML
  - skalowanie, *Patrz:* projektowanie skalowalne
  - szablon, 7, 232

strzałka, 228

styl

- lokalny, 25, 51
- osadzony, 25, 51
- przesłanianie, 25
- zewnętrzny, 25, 51
- źródło, 50

**T**

- tabela, 5
  - CSS3, 177
- tekst, 108, *Patrz też:* font
  - cień, 254
  - jednostki
    - bezwzględne, 113
    - względne, 114
  - kwerendy, 221
  - obrócony, 263
  - stylizacja, 130, 131, 143, 144, 146, 147, 148
    - w siatce, 130, 131, 146
  - wielkość, 113
  - właściwości, 117
- tło, 93, 95, 96, 97, 99, 100, 101, 102, 104, 295
- trójkąt, 228, 267
- Typekit Adobe, 127
- typografia, 130, 131, 145
  - klasyczna, 141

**W**

- walidacja, 3
- warstwa, 93, 94
- wartość
  - koloru, 55, 57
  - liczbowa, 55, 56
  - słowna, 55
- Web Inspector, 294
- właściwość
  - animacja, 222

background, 94, 101  
background-attachment, 94, 100  
background-break, 94, 102  
background-clip, 94, 102, 194  
background-color, 94, 95  
background-image, 94, 95  
background-origin, 94, 102  
background-position, 94, 97  
background-repeat, 94, 96  
background-size, 94, 99  
border, 63  
bottom, 87  
box-sizing, 168, 180  
clear, 75, 76, 81, 82  
clear:both, 160  
clear:left, 160  
display, 10, 93, 225, 245, 295  
float, 75, 76, 78  
font, 109, 117  
font-family, 109, 145  
font-size, 109, 112  
font-style, 109, 115  
font-variant, 109, 116, 145  
font-weight, 109, 116  
left, 87  
letter-spacing, 118, 119, 145  
line-height, 118, 123, 145, 148  
opacity, 247, 248, 287  
overflow, 79  
position, 86, 87, 88, 89, 90, 194  
right, 87  
table, 172, 177  
table-cell, 178  
text-align, 118, 122, 244, 245, 271  
text-decoration, 118, 122  
text-indent, 118, 119  
text-transform, 118, 124  
tła, 94  
top, 87

transform, 103, 263  
transform, 264  
transform-origin, 264  
vertical-align, 118, 125  
visibility, 248, 287, 288  
word-spacing, 118, 121, 145  
word-wrap, 171  
zbiorca, 64  
z-index, 224, 227  
Wroblewski Luke, 201, 272  
wypełnienie, 297

## Y

ySlow, 166

## Z

zaokrąglone rogi, 240  
zapytanie medialne, 274, 276, 281, 285  
znacznik, 1, 2, *Patrz też:* element  
a, 9, 93  
abbr, 20  
alt, 4  
article, 1, 2, 11, 44, 182, 263  
aside, 1  
atrybut, 4  
alt, 4  
charset, 8  
class, 4, 28, 35  
href, 9  
id, 4, 28, 35, 38  
placeholder, 201  
src, 4, 9  
blockquote, 18, 19  
blokowy, 4  
body, 7, 8, 16, 293  
cite, 18  
div, 2, 57, 81, 90, 91, 92, 119, 163, 164, 165,  
166, 168, 171, 178, 179, 180, 182, 184,

## znacznik

- div main\_wrapper, 176
- div threecolwrap, 176
- DOCTYPE, 7
- em, 6, 20
- footer, 1, 44, 159, 167, 168, 233, 268
- główny, *Patrz:* znacznik html
- h1, *Patrz:* nagłówek h1
- head, 7, 8
- header, 1, 233, 236, 238
- html, 7
- img, 4, 9, 13, 93
- input, 201
- li, 5, 6, 187, 194, 271
- liniowy, 4, 19, 20
- link, 25, 277
- listy, 5
- meta, 8, 278
- nav, 1, 39, 44, 49, 176, 180, 182, 187, 234, 244
- nieokalający, 2, 3
- okalający, 2
- ol, 5, 6
- otwierający, 2, 6
- pozbawiony znaczenia semantycznego, 2
- section, 1, 21, 182, 233
- sidebar, 44
- span, 2, 36, 37, 47, 93, 147, 164, 212
- strong, 20
- strukturalny, 1, 21
- style, 26, 274
- tag, 274
- title, 5, 8, 239
- treści, 1
- ul, 5, 39, 40, 186, 189, 190, 191, 193, 194, 197, 200, 236, 242, 245, 246, 258, 271
- zagnieżdżanie, 6, 16, 18
- zamykający, 2, 3, 6
- znak biały, 13, 291

# PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW  
w działający bankomat!

**Dowiedz się więcej i dołącz już dzisiaj!**

<http://program-partnerski.helion.pl>

GRUPA WYDAWNICZA

 **Helion SA**

# Obowiązkowa lektura dla każdego webmastera!

Kaskadowe arkusze stylów (CSS) to technologia, która pozwoliła oddzielić treść od formy jej prezentacji. Dzięki temu tworzenie stron stało się prostsze i przyjemniejsze. Uzyskanie spójnego wyglądu witryny i błyskawiczne wprowadzanie zmian graficznych w obrębie całej strony nie byłoby możliwe bez stylów CSS. Kolejna wersja — CSS3 — dostarcza jeszcze więcej możliwości. Atrakcyjne efekty wizualne czy obsługa wielu formatów ekranu to tylko niektóre z nich.

Kolejne wydanie tej książki zostało ulepszone, poprawione i zaktualizowane o nowe funkcje wersji CSS3. W trakcie lektury nauczysz się precyzyjnie pozycjonować elementy, ustawiać marginesy, umieszczać obrazy w tle oraz tworzyć eleganckie tabele. Ponadto zobaczysz, jak przygotować atrakcyjny formularz, menu lub listę. Twoją szczególną uwagę z pewnością zwrócą rozdziały poświęcone CSS3. Oszalałające efekty specjalne, przystosowanie do obsługi różnych formatów ekranu oraz wsparcie dla urządzeń mobilnych to tylko część czekających na Ciebie atrakcji. Książka ta jest doskonałym kompendium wiedzy na temat kaskadowych arkuszy stylów — warto w nią zainwestować!

Sięgnij po tę książkę i zdobądź wiedzę na temat:

- podstaw arkuszy stylów CSS
- zaawansowanych metod selekcji atrybutów
- nowości w CSS3
- tworzenia nowoczesnych i elastycznych stron WWW

**New Riders** VOICES THAT MATTER™



**Helion**

Helion SA  
ul. Kościuszki 1c, 44-100 Gliwice  
tel.: 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
<http://helion.pl>

**helion.pl**  
księgarnia  
internetowa

Cena: 59,00 zł

ISBN 978-83-246-7066-6

Sprawdź najnowsze promocje:  
• <http://helion.pl/promocje>  
Książki najchętniej czytane:  
• <http://helion.pl/bestsellery>  
Zamów informacje o nowościach:  
• <http://helion.pl/nowosci>



9 788324 670666

Nr katalogowy: 13891

Księgarnia internetowa:  
<http://helion.pl>

Zamówienia telefoniczne:  
0 801 339900

0 601 339900

Informatyka w najlepszym wydaniu

creative  
available  
on edge

PEARSON