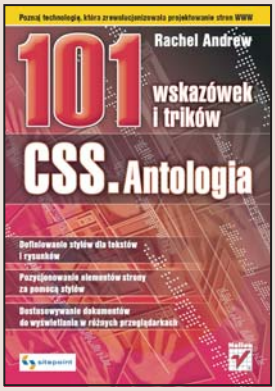
CSS. Antologia. 101 wskazówek i trików



Autor: Rachel Andrew
Tłumaczenie: Krzysztof Jurczyk
ISBN: 83-7361-966-6
Tytuł oryginału: The CSS Anthology: 101 Essential Tips, Tricks and Hacks
Format: B5, stron: 336

Poznaj technologię, która zrewolucjonizowała projektowanie stron WWW

- Definiowanie stylów dla tekstów i rysunków
- Pozycjonowanie elementów strony za pomocą stylów
- Dostosowywanie dokumentów do wyświetlania w różnych przeglądarkach

Kaskadowe arkusze stylów (CSS) całkowicie zmieniły sposób projektowania stron WWW. Formatowanie elementów za pomocą znaczników, żmudne rozmieszczanie ich w komórkach tabel oraz sztuczki, dzięki którym uzyskiwano efekt podświetlenia, to techniki uważane dziś już za przestarzałe. Wykorzystując technologię CSS możemy definiować wygląd każdego elementu umieszczonego na stronie. Przystosowanie strony do wyświetlania na ekranie telefonu komórkowego lub urządzenia PDA również nie stanowi problemu, jeśli zastosujemy do tego kaskadowe arkusze stylów. Opanowanie tej technologii jest dziś nieodzownym elementem wiedzy każdego projektanta witryn WWW.

Książka „CSS. Antologia. 101 wskazówek i trików" to zbiór porad dotyczących stosowania kaskadowych arkuszy stylów. Wykorzystując opisane w książce sposoby, zaprojektujesz estetyczne i efektywne witryny WWW i aplikacje internetowe. Czytając tę książkę, dowiesz się, jak definiować style dla elementów tekstowych i graficznych, w jaki sposób formatować tabele i formularze oraz jak stworzyć podświetlane przyciski bez korzystania z JavaScriptu. Przeczytasz tu również o pozycjonowaniu elementów strony za pomocą stylów, dostosowywaniu strony do wyświetlania w różnych przeglądarkach, systemach operacyjnych i urządzeniach. Poznasz także techniki umożliwiające uzyskanie „efektów specjalnych", które uatrakcyjnią wygląd Twojej witryny.

- Stosowanie stylów dla tekstu
- Przypisywanie stylów do elementów graficznych
- Budowanie mechanizmów nawigacyjnych
- Formatowanie tabel i formularzy
- Zachowanie zgodności witryny z różnymi modelami przeglądarek
- Definiowanie układu strony WWW za pomocą stylów
- Tworzenie efektów specjalnych

Odkryj możliwości kaskadowych arkuszy stylów

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

Spis treści

Rozdział 2.

Style tekstowe i inne techniki podstawowe

Ten rozdział omawia zastosowanie CSS do formatowania tekstu, a także dostarcza kolejnej porcji wiedzy na temat CSS oraz odpowiedzi na niektóre z często pojawiających się pytań. Jeśli CSS jest dla Ciebie nową techniką, analizując zawarte tu przykłady poznasz właściwości i sposoby jej wykorzystania. Dzięki nim będziesz mógł szerzej spojrzeć na kaskadowe arkusze stylów i rozpocząć własne eksperymenty. Dla tych, którzy znają już podstawy CSS, niniejszy rozdział będzie przypomnieniem tych wiadomości, które być może zdążyły już ulecieć z pamięci.

Zaprezentowane przykłady działają poprawnie w różnych przeglądarkach w różnych wersjach. Pamiętaj, że testowanie kodu w wielu aplikacjach jest bardzo ważne, chociaż akurat w przypadku stylów tekstowych nie powinno być jakichkolwiek problemów. Jedynie w starszych wersjach przeglądarek mogą wystąpić pewne rozbieżności w wyświetlaniu stron, jednak nie będą to poważne błędy.

Jak zastąpić znacznik czcionki za pomocą CSS

Określanie wyglądu tekstu w przeglądarkach za pomocą CSS jest możliwe od czasu czwartych wersji IE i Netscape, więc nie ma powodów, aby do tego celu wciąż stosować znacznik `<font>`. Zastąpienie go odpowiednim stylem to najprostszy sposób formatowania tekstu przy użyciu CSS.

Korzystając ze znaczników `<font>`, należy koniecznie zdefiniować styl dla każdego akapitu na każdej stronie witryny:

```
<p><font color="#800080"
     face="Verdana,Geneva,Arial,Helvetica,sans-serif">Nadziewane papryki to
     znakomita przystawka lub dodatek do dania kuchni chińskiej. Nadają się również
     do umieszczenia na stole szwedzkim i nawet dzieci lubią je jeść. </font></p>
```

Rozwiązanie

Korzystając z kaskadowych arkuszy stylów, należy zdefiniować w stylu właściwość `color` znacznika `<p>` jako `#800080`, a właściwość `font-family` jako `Verdana,Geneva, Arial,Helvetica,sans-serif`:

```
p {
  color: #800080;
  font-family: Verdana,Geneva,Arial,Helvetica,sans-serif;
}
```

Od tej pory każdy tekst zawarty w znaczniku `<p>` będzie wyświetlany zgodnie z powyższym stylem, dzięki czemu nie będziesz musiał wielokrotnie stosować dodatkowych opisów do każdego znacznika. Znacznie łatwiej będzie Ci zmodyfikować witrynę, gdy klient niespodziewanie zażąda zmiany w stu dokumentach czcionki z Verdana na Times!

Jak opisywać wielkość czcionki — za pomocą pikseli, punktów, emów czy innej jednostki

W CSS wielkość tekstu można określić za pomocą właściwości `font-size`. Na przykład:

```
font-size: 12px;
```

Istnieje jednak szereg innych sposobów definiowania rozmiarów czcionki. Decyzja, który wybrać, wymaga podstawowej wiedzy o względnej wartości każdej z dostępnych jednostek rozmiaru tekstu.

Rozwiązanie

Jednostki rozmiaru tekstu

W tabeli 2.1 zebrane zostały jednostki, które można stosować do definiowania rozmiaru tekstu w dokumentach.

Tabela 2.1. *Jednostki rozmiaru tekstu*

Identyfikator jednostki	Nazwa jednostki
`pt`	punkt
`pc`	pica
`px`	piksel
`em`	em
`ex`	ex
`%`	procent

Punkty i pice

```
p {
  font-size: 10pt;
}
```

Powinieneś unikać stosowania punktów i pic do definiowania rozmiaru tekstu wyświetlanego na ekranie. Jednostki te znakomicie nadają się do określania rozmiaru w drukarstwie, gdyż do tego celu zostały stworzone. Punkt jest równy $^1/_{72}$ cala, a pica $^1/_6$ cala. Rozmiar wydrukowanego tekstu będzie dokładnie odpowiadał podanym wartościom. Niestety komputery nie potrafią dokładnie określić rzeczywistej wielkości elementów wyświetlanych na ekranie monitora, dlatego zgadują — często niepoprawnie — rozmiary punktu lub picy, przez co końcowy efekt różni się w zależności od platformy. Jeśli tworzysz drukowany arkusz stylów lub dokument przeznaczony do wydrukowania, możesz stosować opisywane jednostki. Zasadniczo jednak należy ich unikać podczas projektowania stron internetowych.

Piksele

```
p {
  font-size: 12px;
}
```

Wielu programistów do określania wielkości tekstu woli stosować piksele. Jednostka ta pozwala zachować jednolity wygląd strony w różnych przeglądarkach i na różnych platformach. Z drugiej strony piksele nie uwzględniają ustawień użytkownika w przeglądarce, a ponadto w większości przeglądarek użytkownik nie może skalować tekstu, którego rozmiar określono w pikselach. To poważny problem dla tych osób, które muszą powiększać wyświetlany na ekranie tekst, aby móc go odczytać. Z tego względu, mimo że piksele wydają się najprostszym rozwiązaniem, powinno się unikać ich stosowania wszędzie tam, gdzie dostępna jest inna metoda określania rozmiaru czcionki — zwłaszcza w przypadku stron z dużą ilością tekstu. Jeśli tworzysz dokument przeznaczony do wydrukowania lub używasz do drukowania arkuszy stylów, powinieneś całkowicie zrezygnować z określania rozmiaru tekstu w pikselach. Piksele nie mają żadnego znaczenia w drukarstwie i — podobnie jak przy wyświetlaniu na ekranie tekstu określonego w punktach — aplikacje drukujące próbują jedynie odgadnąć wielkość czcionki, jaka powinna pojawić się na papierze.

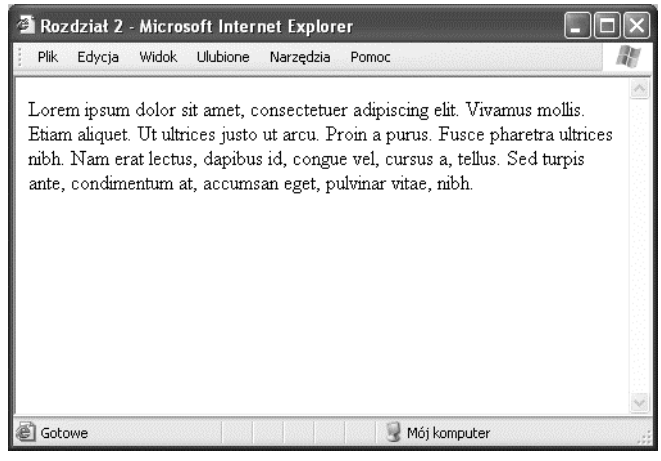
Emy

Em jest względną jednostką rozmiaru tekstu — 1 em jest równy wysokości litery „M” w domyślnym rozmiarze czcionki. Jeśli chodzi o CSS, `1em` jest równy domyślnej wielkości czcionki użytkownika lub — jeżeli różni się on od czcionki domyślnej — równy wielkości elementu nadrzędnego. Jeśli stosujesz emy (lub inną jednostkę względną) do oznaczania wszystkich rozmiarów czcionek w dokumencie, użytkownicy będą mogli zmieniać rozmiar wyświetlanego tekstu zgodnie z własnymi preferencjami. Zadeklarujmy rozmiar tekstu w znaczniku <p> jako `1em`:

```
p {
  font-size: 1em;
}
```

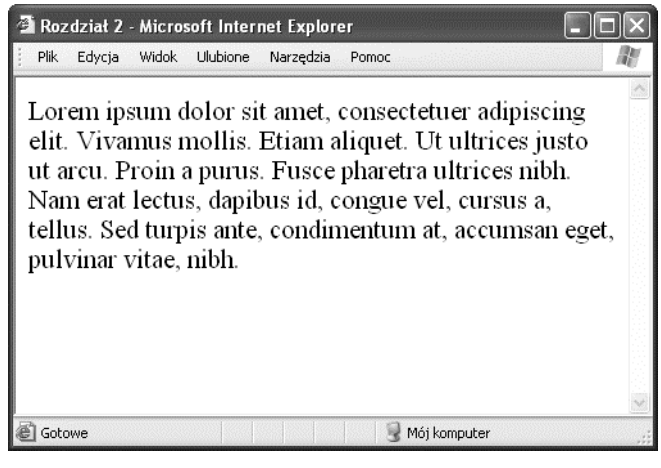
Użytkownik korzystający z przeglądarki Internet Explorer 6, w której rozmiar wyświetlanego tekstu został ustawiony jako *Średni*, ujrzy akapit w postaci przedstawionej na rysunku 2.1.

Rysunek 2.1. *Wygląd okna przeglądarki internetowej przy ustawieniu atrybutu font-size na 1 em, a rozmiaru czcionki w przeglądarce na Średni*



Jeżeli natomiast użytkownik będzie miał przeglądarkę skonfigurowaną tak, aby czcionka wyświetlana była jako *Największa*, wówczas tekst o wielkości 1 em będzie wyglądał tak, jak przedstawia to rysunek 2.2.

Rysunek 2.2. *Wygląd okna przeglądarki internetowej przy ustawieniu atrybutu font-size na 1 em, a rozmiaru czcionki w przeglądarce na Największa*



Z punktu widzenia projektanta taki sposób definiowania rozmiaru tekstu zmniejsza kontrolę nad wyglądem strony w przeglądarce użytkownika. Z drugiej strony umożliwia on przeczytanie zawartości przez tych użytkowników, którzy potrzebują dużej czcionki.

Wartości emów mogą być przedstawione jako liczby dziesiętne. Na przykład, aby wyświetlić tekst o rozmiarze o 10% mniejszym niż domyślny rozmiar czcionki użytkownika (lub 10% mniejszym niż rozmiar elementu nadrzędnego), powinieneś napisać:

```
p {
  font-size: 0.9em;
}
```

Natomiast aby wyświetlić tekst o 10% większy:

```
p {
  font-size: 1.1em;
}
```

Exy

Ex jest względną jednostką wielkości odpowiadającą wysokości małej litery „x" o domyślnym rozmiarze. Teoretycznie, jeżeli zdefiniujesz rozmiar czcionki jako 1 ex, wielkie litery w wyświetlanym tekście powinny mieć tę samą wysokość, co „x" w przypadku braku zdefiniowanej wielkości czcionki.

Niestety dzisiejsze przeglądarki nie posiadają funkcji typograficznych umożliwiających dokładne określenie rozmiaru ex i zazwyczaj w jakiś sposób go szacują. Z tego powodu jednostka ta rzadko bywa stosowana w projektowaniu stron internetowych.

Procenty

```
p {
  font-size: 100%;
}
```

Podobnie jak w przypadku emów i exów, rozmiar czcionki wyrażony w procentach uwzględnia konfigurację przeglądarki użytkownika i tak opisane czcionki mogą być dowolnie skalowane. Ustawienie znacznika <p> na 100% spowoduje wyświetlenie tekstu o wielkości równej domyślnym ustawieniom czcionki użytkownika (podobnie jak dla 1 em). Zmniejszenie wartości procentowej spowoduje pomniejszenie wyświetlanego tekstu:

```
p {
  font-size: 90%;
}
```

Zwiększenie wartości spowoduje powiększenie tekstu:

```
p {
  font-size: 150%;
}
```

Słowa kluczowe

Rozmiar tekstu można również definiować za pomocą słów kluczowych.

Słowa kluczowe definiujące bezwzględny rozmiar czcionki

W CSS istnieje siedem słów kluczowych określających bezwzględny rozmiar tekstu:

- ♦ `xx-small`,
- ♦ `x-small`,
- ♦ `small`,

- `medium`,
- `large`,
- `x-large`,
- `xx-large`.

Słowa te są względne wobec siebie nawzajem, a przeglądarki mogą interpretować je na różne sposoby. Większość przeglądarek wyświetla rozmiar `medium` jako odpowiadający rozmiarowi tekstu bez zdefiniowanego stylu, natomiast pozostałe rozmiary są odpowiednio skalowane. Wyjątek stanowi przeglądarka Internet Explorer 5 (oraz wersja 6 w przypadku niektórych typów dokumentów), w której rozmiarowi tekstu bez zdefiniowanego stylu odpowiada słowo kluczowe `small`.

Bezwzględny rozmiar czcionki oznacza w tym przypadku, że nie jest on zależny od żadnego elementu nadrzędnego. W przeciwieństwie jednak do wartości bezwzględnych podawanych w pikselach i punktach, umożliwia on zmianę rozmiaru oglądanego przez użytkownika tekstu oraz uwzględnia ustawienia przeglądarki. Głównym problemem w stosowaniu słów kluczowych jest to, że tekst o rozmiarze `x-small` może być w jednej przeglądarce doskonale czytelny, a w innej zdecydowanie za mały.

Słowa kluczowe definiujące względny rozmiar czcionki

Stosując słowa kluczowe `larger` lub `smaller`, definiujemy względny rozmiar czcionki, który określany będzie na podstawie rozmiaru elementu podrzędnego — podobnie jak w przypadku zastosowania jednostki em lub %. Jeżeli zatem zdefiniujesz znacznik `<p>` jako `small` i zechcesz wyświetlić wyróżniony tekst jako odpowiednio większy, powinieneś dodać do arkusza stylów następujący fragment (listing 2.1):

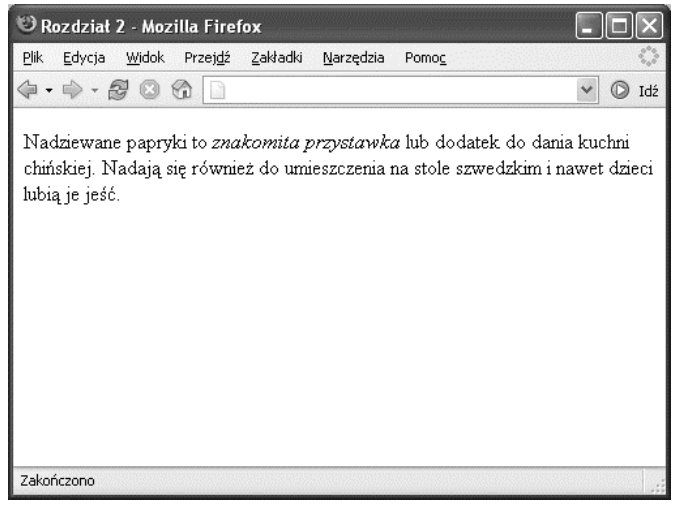
Listing 2.1. *Zawartość pliku wzgledny.css*

```
p {
  font-size: small;
}
em {
  font-size: larger;
}
```

Poniższy fragment zostanie wyświetlony w przeglądarce tak, jak zostało to pokazane na rysunku 2.3, ponieważ tekst zawarty w znaczniku `<em>` jest zdefiniowany jako większy (`larger`) niż jego element nadrzędny — znacznik `<p>` (listing 2.2).

Listing 2.2. *Zawartość pliku wzgledny.html (fragment)*

```
<p>Nadziewane papryki to <em>znakomita przystawka</em> lub dodatek
   do dania kuchni
   chińskiej. Nadają się również do umieszczenia na stole szwedzkim
   i nawet dzieci lubią je jeść.</p>
```



Rysunek 2.3. *Rozmiar wyróżnionego tekstu jest większy niż pozostałej części akapitu*

Rozmiary względne i ich dziedziczenie

Gdy stosujesz jeden ze sposobów względnego określania rozmiaru czcionki, pamiętaj, że modyfikowany element dziedziczy rozmiar początkowy od elementu podrzędnego, po czym odpowiednio modyfikuje swoją wielkość. Nie stanowi to problemu w przypadku układów, w których nie ma wielu zagnieżdżeń. Jeśli jednak strona zawierać będzie tabele z rozbudowanymi zagnieżdżeniami, wówczas nie zawsze będzie oczywiste, który element jest nadrzędny i dziedziczenie rozmiarów może przebiegać niezgodnie z oczekiwaniami. Ilustruje to poniższy przykład.

Arkusz stylów zawiera następujący kod, definiujący znacznik `td` tak, aby wyświetlać tekst o rozmiarze 80%. Jest to nieco mniejsza wartość od domyślnej czcionki użytkownika i może być modyfikowana za pomocą ustawień przeglądarki (listing 2.3):

Listing 2.3. *Plik zagniezdzanie.css*

```
td {
  font-size: 80%;
}
```

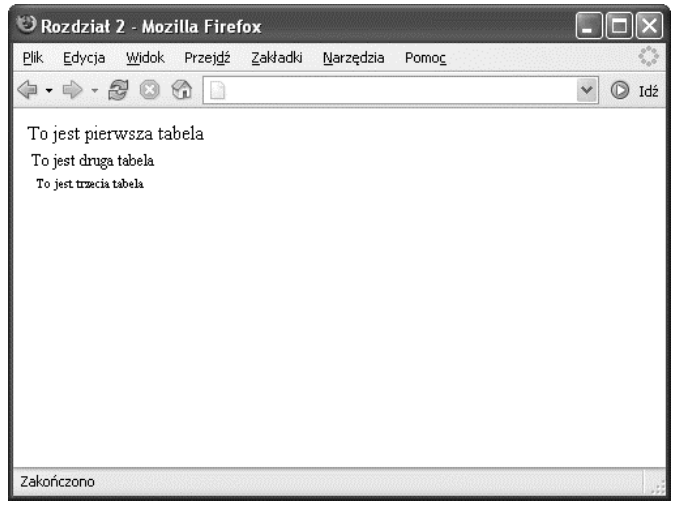
Na stronie, która nie posiada zagnieżdżonych komórek tabel, tekst będzie konsekwentnie wyświetlany jako pomniejszony. Jednak w przypadku układu zawierającego zagnieżdżone tabele (listing 2.4), rozmiar tekstu w każdej z tych tabel wyniesie 80% rozmiaru czcionki w tabeli nadrzędnej.

Listing 2.4. *Plik zagniezdzanie.html (fragment)*

```
<table>
  <tr>
    <td>To jest pierwsza tabela
      <table>
```

```
        <tr>
          <td> To jest druga tabela
            <table>
              <tr>
                <td> To jest trzecia tabela</td>
              </tr>
            </table>
          </td>
        </tr>
      </table>
    </td>
  </tr>
</table>
```

Efekt działania powyższego fragmentu został pokazany na rysunku 2.4. Jak widzisz, w każdej kolejnej tabeli tekst staje się mniejszy.



Rysunek 2.4. *Efekt zastosowania względnych rozmiarów czcionek dla tekstu w tabelach zagnieżdżonych*

Analiza

Przy wyborze sposobu określania rozmiaru tekstu należy zawsze zastosować metodę, która pozwoli użytkownikom zmieniać rozmiar wyświetlanej czcionki i dzięki której rozmiar ten będzie zgodny z ustawieniami przeglądarki użytkownika. Rozmiary względne wydają się funkcjonować poprawnie w układach CSS i prostych układach zawierających tabele, jednak w przypadku tabel zagnieżdżonych mogą sprawić trochę kłopotów ze względu na sposób dziedziczenia rozmiaru. Jeśli jesteś zdecydowany, aby zastosować jednostki bezwzględne, rozważ włączenie do interfejsu użytkownika opcji wyboru rozmiaru wyświetlanego tekstu.

Jak zdefiniować czcionkę dla wyświetlanego tekstu

Rozwiązanie

Zdefiniuj czcionkę, za pomocą której ma być wyświetlany tekst akapitu, korzystając z właściwości `font-family`:

```
p {
  font-family: Verdana;
}
```

Analiza

Oprócz najbardziej popularnych czcionek, takich jak Verdana czy Times New Roman, CSS umożliwia stosowanie innych znanych rodzin czcionek:

- `serif`,
- `sans-serif`,
- `monospace`,
- `cursive`,
- `fantasy`.

Przy definiowaniu czcionek należy pamiętać, że użytkownicy prawdopodobnie nie mają zainstalowanych tych samych krojów co my. Jeżeli czcionka nie zostanie odnaleziona w systemie, wówczas tekst będzie wyświetlany za pomocą domyślnej czcionki przeglądarki internetowej, bez względu na układ witryny.

Najprostsze rozwiązanie to zadeklarować popularną rodzinę czcionek i pozostawić wybór konkretnej czcionki systemowi użytkownika. Na przykład, jeżeli dokument ma być wyświetlony przy użyciu czcionki sans-serif (Arial), wystarczy napisać następującą deklarację:

```
p {
  font-family: sans-serif;
}
```

Z pewnością jednak chciałbyś mieć większą kontrolę nad sposobem wyświetlania tekstu w przeglądarce. Na szczęście istnieje możliwość zdefiniowania konkretnej nazwy czcionki, a nie tylko rodziny czcionek. Przyjrzyjmy się poniższej deklaracji znacznika <p>. Zgodnie z nią najpierw sprawdzona zostanie obecność w systemie czcionki Verdana; w przypadku jej braku przeglądarka sprawdzi obecność czcionki Geneva. Jeśli przeglądarka także tej nie odnajdzie, nastąpi próba odszukania czcionek Arial i Helvetica.

Jeśli te również nie zostaną znalezione, tekst zostanie wyświetlony za pomocą domyślnej czcionki systemowej sans-serif.

```
p {
  font-family: Verdana, Geneva, Arial, Helvetica, sans-serif;
}
```

Jak usunąć podkreślenie z łączy do stron internetowych

Domyślnie tekst będący łączem do innej strony WWW jest podkreślony i różni się kolorem od reszty tekstu. Czasem jednak będziesz chciał to wyróżnienie usunąć.

Rozwiązanie

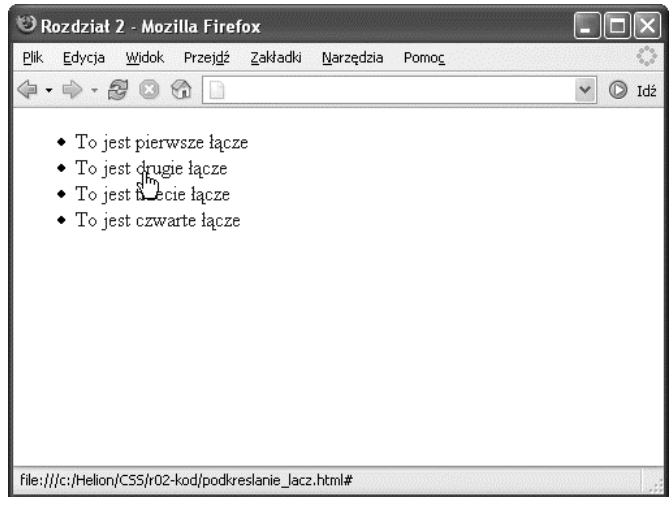
Aby usunąć podkreślenie, należy zastosować właściwość `text-decoration`. Domyślna wartość tej właściwości dla znacznika `<a>` to `<underline>`. Aby zlikwidować podkreślenie, należy do definicji znacznika dołączyć poniższą linię kodu:

```
text-decoration: none;
```

Efekt widoczny na rysunku 2.5 został uzyskany za pomocą kodu CSS przedstawionego na listingu 2.5.

Listing 2.5. *Plik podkreslanie_lacz.css*

```
a:link, a:visited {
  text-decoration: none;
}
```



Rysunek 2.5. *Modyfikując właściwość text-decoration, można usunąć podkreślenie z łączy internetowych*

Analiza

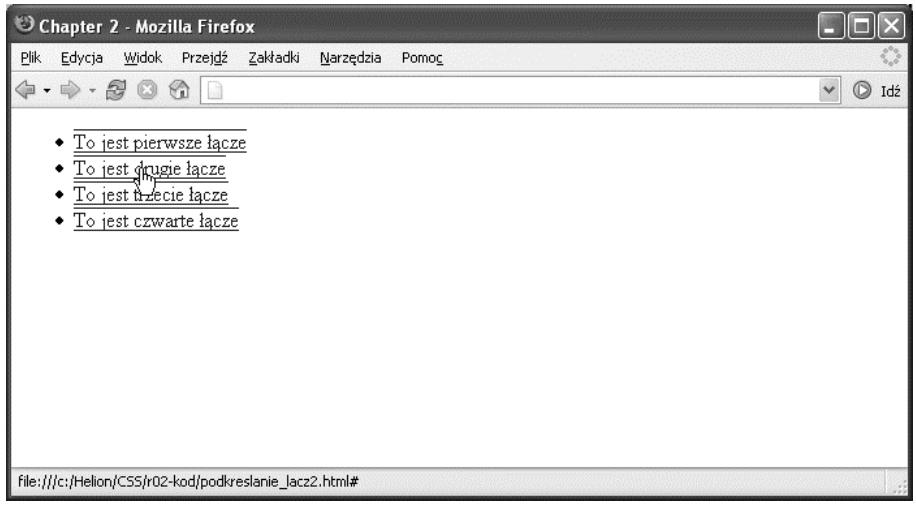
Oprócz `underline` i `none`, właściwość `text-decoration` może przyjmować następujące wartości:

- `overline`,
- `line-through`,
- `blink`.

Wypróbuj działanie tych wartości — zarówno pojedynczo, jak i kilku naraz. Na przykład, aby uzyskać podkreślenie łącza oraz linię nad nim (jak na rysunku 2.6), powinieneś użyć kodu przedstawionego na listingu 2.6.

Listing 2.6. *Plik podkreslanie_lacz2.css*

```
a:link, a:visited {
  text-decoration: underline overline;
}
```



Rysunek 2.6. *Właściwości text-decoration można przypisać kilka wartości naraz, uzyskując w ten sposób interesujące efekty*

Nadużywanie podkreślenia

Istnieje również możliwość zastosowania właściwości `text-decoration` dla tekstu, który nie jest łączem, jednak w takim przypadku należy zachować szczególną ostrożność. Podkreślanie łącz jest przyjęte i stosowane na całym świecie, tak więc użytkownik może potraktować każdy podkreślony tekst jako łącze.

Kiedy nie należy usuwać podkreślenia?

Podkreślanie łączy standardowo obsługują wszystkie przeglądarki internetowe. Każdy użytkownik spodziewa się, że łącza na stronie będą podkreślone. Usunięcie podkreślenia bardzo utrudni ich odnalezienie — nawet jeśli będą innego koloru. Tak więc nie polecam usuwania podkreślenia łączy. Istnieje wiele innych sposobów ich atrakcyjnego wyróżnienia, które nie wymagają rezygnacji ze stosowania podkreśleń.

Zupełnie inny przypadek stanowią łącza w ramach menu lub w formie przycisku. W takiej sytuacji możesz zdecydować się na usunięcie podkreślenia, gdyż jest oczywiste, że nie jest to zwykły tekst.

Jak utworzyć łącze, którego kolor zmieni się po wskazaniu kursorem

Jednym z atrakcyjnych efektów dla łączy jest zmiana wyglądu w chwili wskazania kursorem myszki. Efekt ten stosuje się zazwyczaj w celu zastąpienia standardowych przycisków nawigacji, jednak można go również wykorzystać do wyróżnienia łączy w dokumencie.

Rozwiązanie

Aby uzyskać wspomniany efekt, zmienimy style pseudoklas `:hover` i `:active` tak, aby różniły się od innych pseudoklas znacznika kotwicy.

Tworzenie efektu rozpoczniemy od następującej deklaracji stylów znacznika łącza (listing 2.7).

Listing 2.7. *Plik podkreslanie_lacz3.css*

```
a:link, a:visited, a:hover, a:active {
  text-decoration: underline;
  color: #6A5ACD;
  background-color: transparent;
}
```

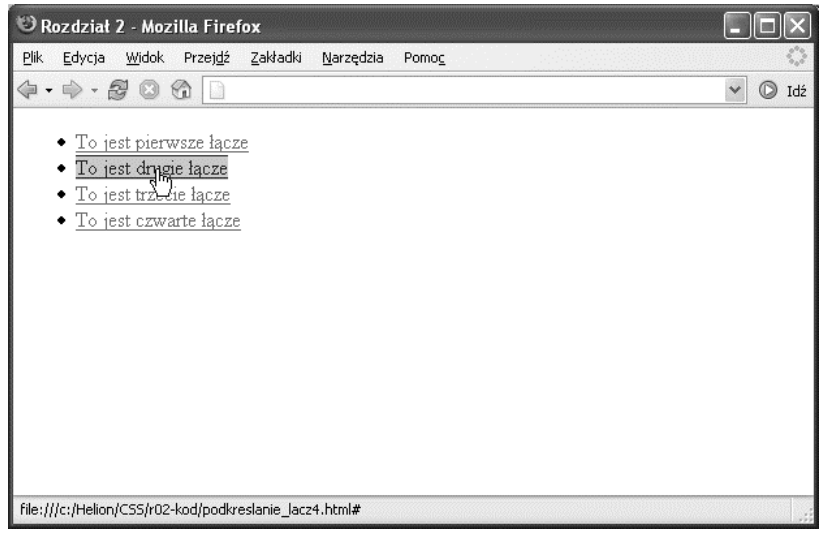
Po zastosowaniu tego stylu łącza będą podkreślone i w kolorze niebieskim `#6A5ACD`, co zostało pokazane na rysunku 2.7.

Aby wyróżnić pseudoklasy `:hover` i `:active`, należy usunąć je z deklaracji innych pseudoklas i wprowadzić ich własne deklaracje. Do standardowego podkreślenia dołączyłam linię górną, zmodyfikowany kolor tła oraz zastosowałam ciemniejszy kolor tekstu. Wynikowy styl CSS został pokazany na listingu 2.8, a efekt końcowy na rysunku 2.8.

Rysunek 2.7. *Dla wszystkich pseudoklas łączy zastosowana została taka sama deklaracja kotwicy*

Listing 2.8. *Plik podkreslanie_lacz4.css*

```
a:link, a:visited {
  text-decoration: underline;
  color: #6A5ACD;
  background-color: transparent;
}
a:hover, a:active {
  text-decoration: underline overline;
  color: #191970;
  background-color: #C9C3ED;
}
```



Rysunek 2.8. *Efekty zastosowania stylu CSS stają się widoczne po wskazaniu łącza kursorem myszki*

Zorientowałeś się zapewne, że w podobny sposób można definiować style dla pozostałych pseudoklas. Przede wszystkim możesz specjalnie wyróżnić łącza do stron, które zostały już odwiedzone przez użytkownika. Aby to zrobić, wystarczy zmodyfikować styl pseudoklasy `:visited`.

Podczas definiowania stylów dla pseudoklas pamiętaj, aby nie zmieniać rozmiaru czy grubości tekstu. Jeśli to zrobisz, zawartość strony będzie „pływać", dostosowując się w chwili wskazania kursorem do powiększonego tekstu.

Kolejność deklaracji pseudoklas

Pseudoklasy łącza powinny zostać zadeklarowane w następującej kolejności: `link`, `visited`, `hover`, `active`. W przeciwnym razie pseudoklasy mogą działać niepoprawnie. Prostym sposobem na zapamiętanie tej kolejności jest mnemonik: LoVeHAte („miłośćnienawiść").

Jak wyświetlić dwa różne style łącza na jednej stronie

W poprzednim przykładzie zobaczyliśmy, jak zdefiniować style dla różnych selektorów znacznika kotwicy. Co jednak zrobić, jeśli chcielibyśmy zastosować różne style łącza w tym samym dokumencie — na przykład bez podkreślenia w menu nawigacyjnym, ale z podkreśleniem w głównej części strony? Jeśli część strony posiada ciemne tło, na pewno chciałbyś w tym miejscu użyć innego koloru łączy. Jak zatem to zrobić?

Rozwiązanie

Zastosowanie wielu różnych stylów łączy zobrazujemy, modyfikując stworzony wcześniej arkusz stylów (listing 2.9).

Listing 2.9. *Plik typy_lacz.css (fragment)*

```
a:link, a:visited {
  text-decoration: underline;
  color: #6A5ACD;
  background-color: transparent;
}

a:hover, a:active {
  text-decoration: underline overline;
  color: #191970;
  background-color: #C9C3ED;
}
```

Mamy zatem zdefiniowany styl domyślny, za pomocą którego wyświetlane będą łącza w naszych dokumentach. Styl ten definiuje niebieski kolor łącza, więc w przypadku niebieskiego tła łącze będzie niewidoczne. Konieczne jest zatem utworzenie drugiego zestawu stylów specjalnie dla łączy na niebieskim tle.

Najpierw należy określić klasę lub identyfikator ID dla obszaru, który zawierać będzie łącza w zmienionym kolorze. Jeśli dla obszaru określono już style CSS, ma on także określoną klasę lub ID, które mogą zostać wykorzystane. Załóżmy, że nasz dokument zawiera kod pokazany na listingu 2.10:

Listing 2.10. *Plik typy_lacz.html (fragment)*

```
<div class="ramka">
   <p>Odwiedź nasz <a href="sklep.html">sklep internetowy</a>, w którym znajdziesz
   wszystko, czego potrzebujesz</p>
</div>
```

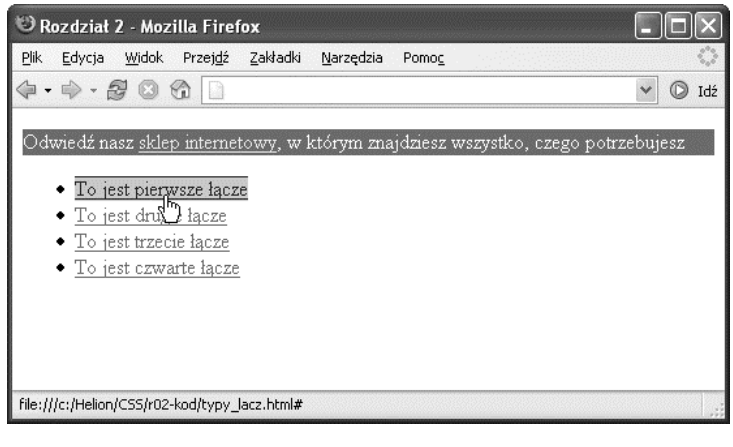
Konieczne jest utworzenie stylu CSS dla łączy w obszarze, w którym element nadrzędny został zmodyfikowany za pomocą klasy `ramka` (listing 2.11):

Listing 2.11. *Plik typy_lacz.css (fragment)*

```
.ramka {
   color: #FFFFFF;
   background-color: #6A5ACD;
}
.ramka a:link, .ramka a:visited {
   text-decoration: underline;
   color: #E4E2F6;
   background-color: transparent;
}

.ramka a:hover, .ramka a:active {
  background-color: #C9C3ED;
  color: #191970;
}
```

Na rysunku 2.9 widoczne są łącza sformatowane za pomocą dwóch różnych stylów — w głównym dokumencie widoczne są efekty pierwszego stylu, natomiast łącze w niebieskiej ramce zostało sformatowane za pomocą drugiego stylu.



Rysunek 2.9. *W jednym dokumencie można wykorzystać dwa różne style formatowania łączy*

Jak dodać kolor tła do nagłówka

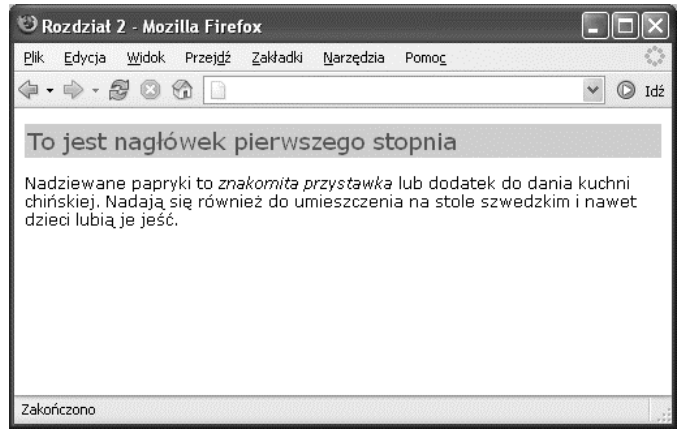
Kolor tła można zdefiniować dla dowolnego elementu strony internetowej, w tym również dla nagłówka.

Rozwiązanie

Na listingu 2.12 przedstawiony został arkusz CSS definiujący styl dla wszystkich nagłówków pierwszego stopnia w dokumencie. Efekt działania tego stylu widoczny jest na rysunku 2.10.

Listing 2.12. *Plik kolor_naglowka.css (fragment)*

```
h1 {
   background-color: #ADD8E6;
   color: #256579;
   font: 18px Verdana, Geneva, Arial, Helvetica, sans-serif;
   padding: 2px;
}
```



Rysunek 2.10. *Nagłówki również mogą posiadać tło*

Miejsce dla koloru

Jeśli dodajemy kolor tła do nagłówka, warto zmodyfikować sam tekst tak, aby utworzyć trochę wolnej przestrzeni pomiędzy nim a krawędzią kolorowego tła — tak jak w nagłówku w powyższym przykładzie.

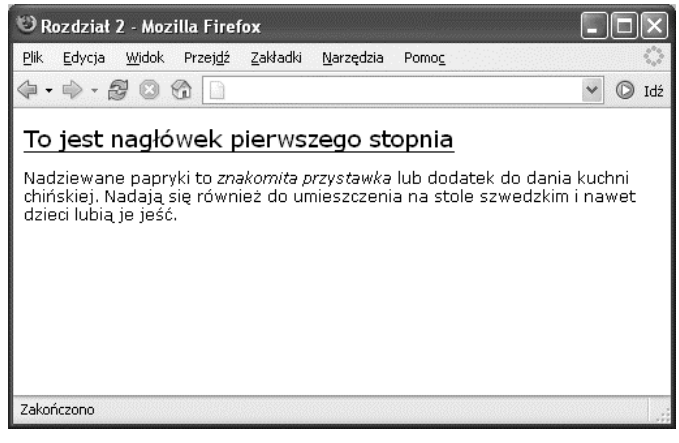
Jak podkreślić tekst nagłówka

Rozwiązanie

Są dwa sposoby, aby dodać podkreślenie do tekstu nagłówka. Najprościej jest zastosować właściwość `text-decoration`, którą poznaliśmy przy okazji nadawania stylów łączom. Metoda ta pozwala podkreślić tekst linią w tym samym kolorze co tekst (listing 2.13 i rysunek 2.11).

Listing 2.13. *Plik podkreslenie_naglowka.css (fragment)*

```
h1 {
   font: 18px Verdana, Geneva, Arial, Helvetica, sans-serif;
   text-decoration: underline;
}
```



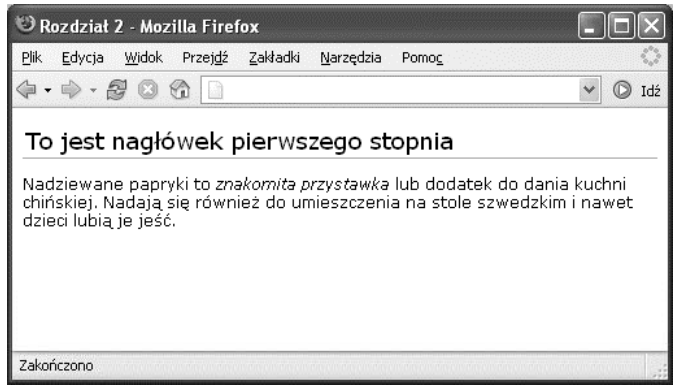
Rysunek 2.11. *Podkreślenie tekstu nagłówka za pomocą właściwości text-decoration*

Inny sposób podkreślenia nagłówka to dodanie do niego dolnej ramki (listing 2.14). Takie rozwiązanie jest bardziej elastyczne — możesz ustalić odległość linii podkreślenia od tekstu, a także zastosować inny kolor linii niż tekstu. Jednak w tym przypadku efekt końcowy może nieznacznie różnić się w poszczególnych przeglądarkach, musisz zatem upewnić się, że przynajmniej w tych najpopularniejszych otrzymany rezultat będzie odpowiadał Twoim oczekiwaniom (patrz rysunek 2.12).

Listing 2.14. *Plik podkreslenie_naglowka2.css (fragment)*

```
h1 {
   font: 18px Verdana, Geneva, Arial, Helvetica, sans-serif;
   padding: 2px;
   border-bottom: 1px solid #aaaaaa;
}
```

Rysunek 2.12. *Podkreślenie tekstu nagłówka z wykorzystaniem dolnej ramki*



Jak usunąć odstęp pomiędzy nagłówkiem <h1> a występującym pod nim akapitem

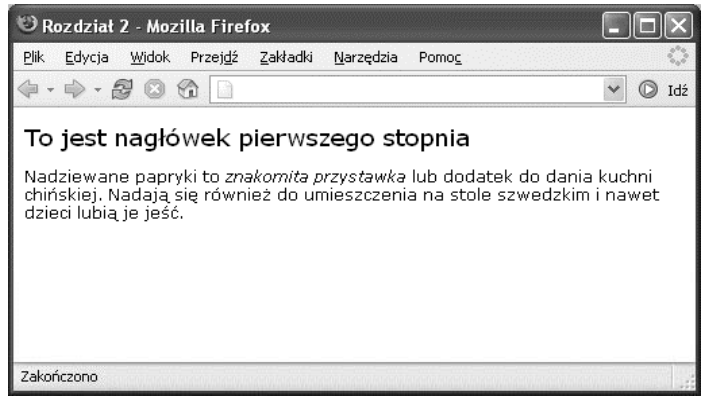
Rozwiązanie

Przeglądarki wyświetlają przerwę pomiędzy nagłówkiem a akapitem, stosując marginesy górny i dolny, które przypisane są domyślnie do znacznika nagłówka i akapitu.

Aby zlikwidować ten odstęp, należy usunąć nie tylko dolny margines nagłówka, ale także górny margines akapitu. Jednak ze względu na to, że niewygodnie byłoby stosować selektor CSS w odniesieniu do „pierwszego akapitu za nagłówkiem", łatwiej będzie przypisać nagłówkowi ujemny margines dolny. Marginesy, w przeciwieństwie do atrybutu `padding` (patrz listing 2.14), mogą przyjmować wartości ujemne.

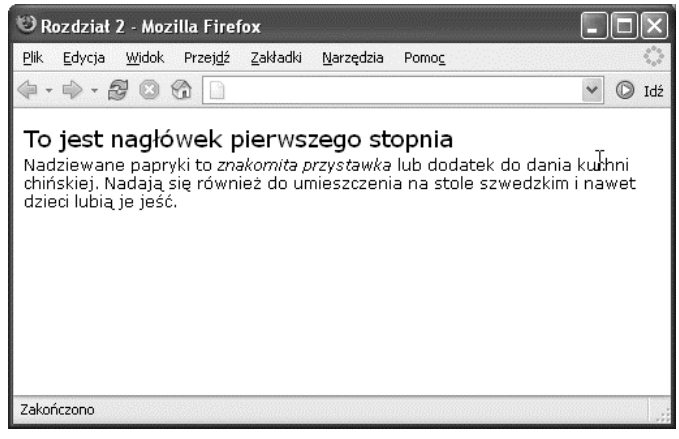
Marginesy widoczne na rysunku 2.13 nie zostały zmienione.

Rysunek 2.13. *Domyślny odstęp pomiędzy nagłówkiem pierwszego stopnia a akapitem*



Na rysunku 2.14 pokazano ten sam nagłówek <h1> zmodyfikowany przez poniższy styl CSS:

```
h1 {
   font: 18px Verdana, Geneva, Arial, Helvetica, sans-serif;
   margin-bottom: -10px;
}
```



Rysunek 2.14. *Sposób wyświetlania nagłówka uległ zmianie po przypisaniu wartości –10px do właściwości margin-bottom*

Jak podświetlić tekst bez stosowania znaczników czcionki

Bez stosowania CSS do podświetlenia fragmentu tekstu (na przykład w celu podkreślenia istotnego wyrażenia lub wyróżnienia wystąpień danej frazy w tekście dokumentu) należałoby wykorzystać odpowiedni znacznik czcionki.

Rozwiązanie

CSS pozwala utworzyć klasę dla stylu podświetlenia, a następnie zastosować ją poprzez umieszczenie tekstu, który ma zostać podświetlony, wewnątrz znaczników dotyczących tej klasy. Na przykład w poniższym akapicie (listing 2.15) fragment tekstu został umieszczony wewnątrz znacznika związanego z klasą zolty.

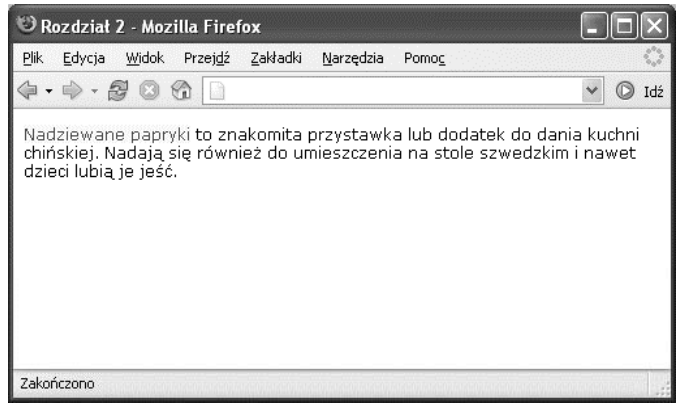
Listing 2.15. *Plik zolty.html (fragment)*

```
<p><span class="zolty">Nadziewane papryki</span> to znakomita przystawka lub
dodatek
do dania kuchni chińskiej. Nadają się również do umieszczenia na stole szwedzkim
i nawet dzieci lubią je jeść.</p>
```

Zawartość klasy zolty została przedstawiona na listingu 2.16, a efekt jej zastosowania widoczny jest na rysunku 2.15.

Listing 2.16. *Plik zolty.css (fragment)*

```
.zolty {
   background-color: #FFFFCC;
   color: #B22222;
}
```



Rysunek 2.15. *Podświetlenie tekstu za pomocą stylu CSS*

Jak zmodyfikować odstęp między liniami tekstu

Jedną z istotnych zalet CSS w porównaniu z klasycznymi znacznikami czcionki jest to, że kaskadowe arkusze stylów oferują o wiele większą kontrolę nad wyglądem tekstu na stronie internetowej. W tym przykładzie zobaczymy, w jaki sposób zmienić odstępy między liniami tekstu.

Rozwiązanie

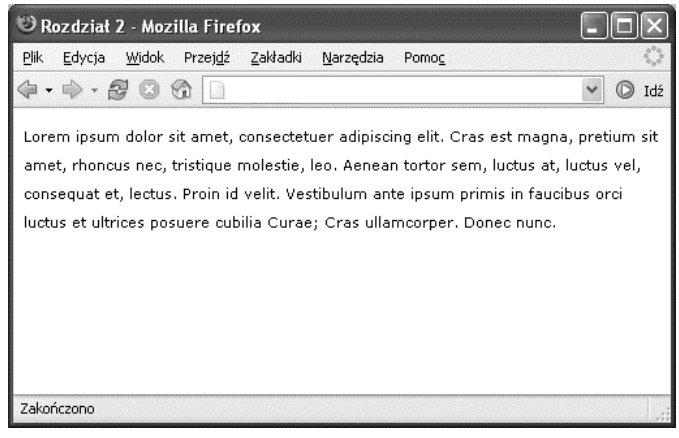
Jeżeli domyślne odstępy pomiędzy liniami są za małe, powiększ je za pomocą właściwości `line-height` (listing 2.17).

Listing 2.17. *Plik odstepy.css*

```
p {
   font: 11px Verdana, Geneva, Arial, Helvetica, sans-serif;
   line-height: 2.0;
}
```

Wynik powyższej modyfikacji znacznika <p> został pokazany na rysunku 2.16.

Nie przesadź ze zbyt dużą odległością między liniami, w przeciwnym razie czytanie tekstu będzie utrudnione.



Rysunek 2.16. *Modyfikacja odstępów między liniami za pomocą właściwości line-height*

Brak jednostek?

Właściwość `line-height` możesz także zdefiniować stosując jednostki wielkości dostępne w CSS, czyli w emach lub pikselach. Jednak w takim przypadku znika zależność pomiędzy wysokością linii a rozmiarem czcionki elementów pochodnych.

Na przykład, jeżeli w powyższym pliku klasa `<span>` definiowałaby duży rozmiar czcionki (`large`), wysokość linii byłaby proporcjonalna do rozmiaru czcionki, gdyż właściwość `line-height` akapitu została ustawiona jako wartość liczbowa `2.0`. Jeżeli natomiast `line-height` miałaby wartość `2 emy` lub `200%`, wówczas klasa `<span>` odziedziczyłaby rzeczywistą wysokość linii, a nie współczynnik proporcjonalności i w rezultacie rozmiar czcionki `large` nie wpłynąłby na wysokość linii w klasie `<span>`. W zależności od efektu, który chcesz uzyskać, takie działanie kodu może być czasami przydatne.

Jak wyjustować tekst

Justowanie tekstu powoduje, że odległość pomiędzy poszczególnymi wyrazami w linii zostaje zmodyfikowana tak, aby tekst został wyrównany zarówno do lewej, jak i do prawej krawędzi — dokładnie tak, w tej książce. Ten sposób wyrównywania tekstu dostępny jest w CSS.

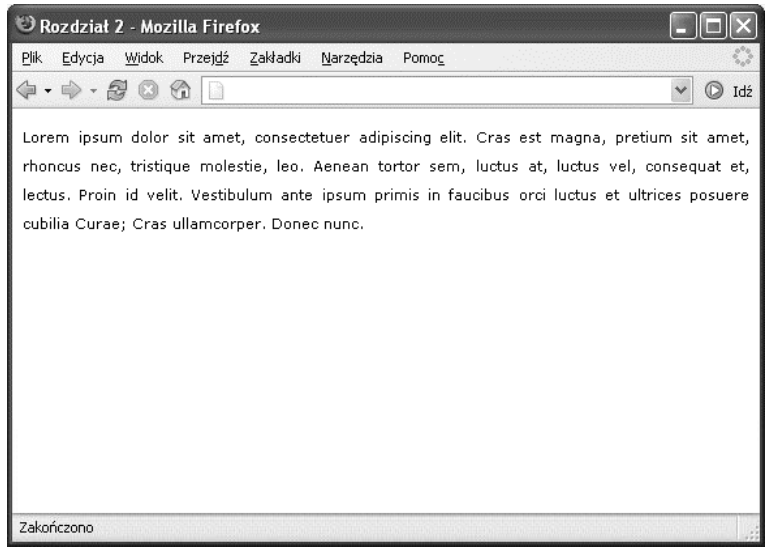
Rozwiązanie

Aby wyjustować tekst akapitu, należy skorzystać z właściwości `text-align` (listing 2.18).

Listing 2.18. *Plik justowanie.css*

```
p {
   text-align: justify;
   font: 11px Verdana, Geneva, Arial, Helvetica, sans-serif;
   line-height: 2.0;
}
```

Na rysunku 2.17 przedstawiony został efekt działania powyższego listingu.



Rysunek 2.17. *Justowanie tekstu za pomocą właściwości text-align*

Analiza

Właściwość `text-align`, obok `justify`, może przyjmować następujące wartości:

`right`

Wyrównuje tekst do prawego marginesu kontenera.

`left`

Wyrównuje tekst do lewego marginesu kontenera.

`center`

Wyśrodkowuje tekst zawarty w kontenerze.

Jak nadać styl poziomej linii

Rozwiązanie

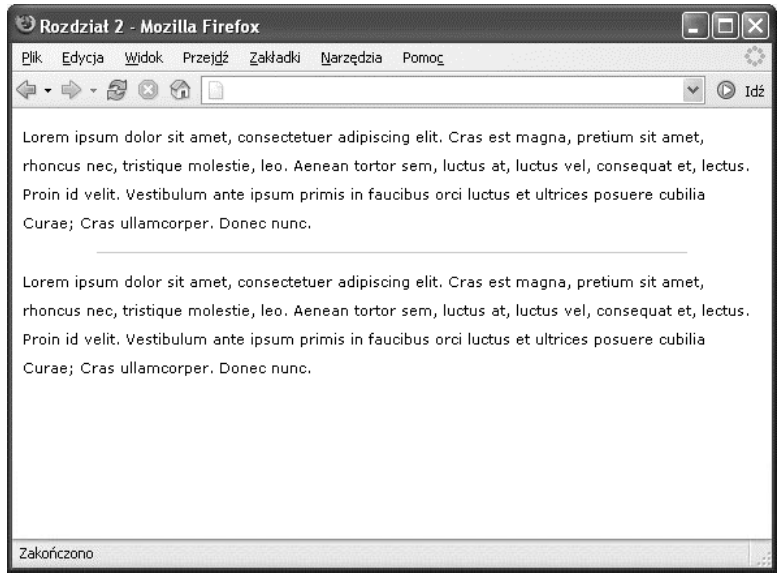
Możesz modyfikować zarówno kolor, jak i wysokość oraz długość linii poziomej. Zwróć uwagę, że w poniższym listingu (2.19) użyłam tych samych wartości dla właściwości `color` oraz `background-color`, ponieważ przeglądarki oparte na Mozilli odczytują wartość `background-color`, podczas gdy Internet Explorer korzysta z właściwości `color`.

Listing 2.19. *Plik styl_hr.css (fragment)*

```
hr {
   border: none;
   background-color: #ADD8E6;
   color: #ADD8E6;
   height: 1px;
   width: 80%;
}
```

Efekt zastosowania powyższego stylu widoczny jest na rysunku 2.18.

Rysunek 2.18. *Zmiana koloru, wysokości i długości linii poziomej*



Jak wprowadzić wcięcie w tekście

Rozwiązanie

Tekst może zostać wcięty przy wykorzystaniu właściwości `padding-left` w odniesieniu do kontenera tekstu (listing 2.20 i 2.21).

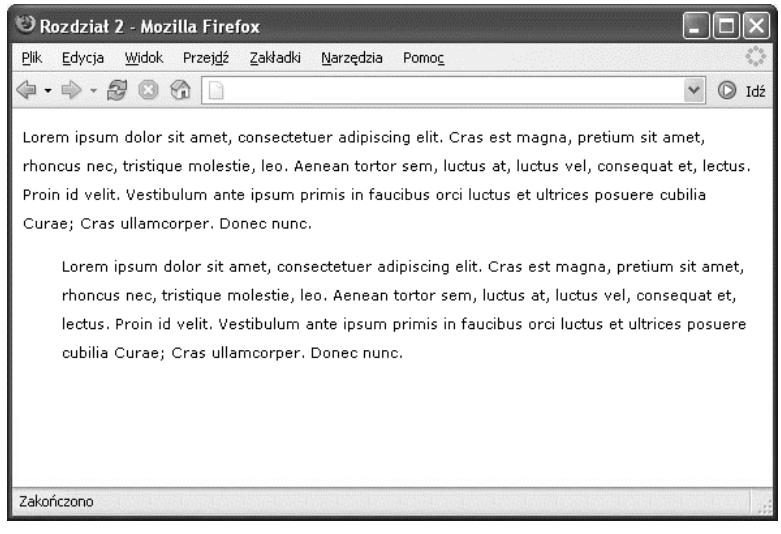
Listing 2.20. *Plik wciecie.html (fragment)*

```
<p class="wciecie">Lorem ipsum dolor sit amet, consectetuer
   adipiscing elit. Vivamus mollis. Etiam aliquet. Ut ultrices
   justo ut arcu. Proin a purus. Fusce pharetra ultrices nibh.
   Nam erat lectus, dapibus id, congue vel, cursus a, tellus.
   Sed turpis ante, condimentum at, accumsan eget, pulvinar
   vitae, nibh.</p>
```

Listing 2.21. *Plik wciecie.css (fragment)*

```
.wciecie {
   padding-left: 30px;
}
```

Akapit, którego tekst został wcięty, pokazany został na rysunku 2.19.



Rysunek 2.19. *Wcięcie tekstu za pomocą właściwości CSS*

Analiza

Aby uzyskać wcięcie tekstu, nie powinno się stosować znacznika HTML `<blockquote>`, chyba że tekst ten rzeczywiście jest cytatem. Mimo że niektóre graficzne narzędzia do tworzenia witryn internetowych, jak na przykład Macromedia Dreamweaver, często opisują ten znacznik jako „tekst wcięty", oprzyj się pokusie stosowania go w tym celu. Zamiast tego zdefiniuj regułę CSS, za pomocą której będziesz mógł stosować wcięcie do odpowiednich akapitów tekstu. Znacznik `<blockquote>` przeznaczony jest do oznaczania cytatu i jest wykorzystywany przez aplikacje dla osób z upośledzeniem wzroku. Oznaczony w ten sposób tekst będzie odczytany z inną intonacją, aby odbiorca mógł się zorientować, że jest to cytat. Jeżeli zastosujesz `<blockquote>` do wcięcia zwykłego akapitu, może to wprowadzić w błąd osoby korzystające z oprogramowania tego typu.

Wcięcie pierwszej linii

Ta sama technika pozwala także zastosować wcięcie tylko w odniesieniu do pierwszej linii każdego akapitu. Aby wciąć pierwszy akapit, należy użyć właściwości `text-indent` w odniesieniu do znacznika akapitu lub do klasy stosowanej do modyfikacji akapitów:

```
p {
   text-indent: 20px;
}
```

Jak wyśrodkować tekst

Rozwiązanie

Aby wyśrodkować tekst lub inny element, należy właściwości `text-align` przypisać wartość `center` (listing 2.22 i 2.23).

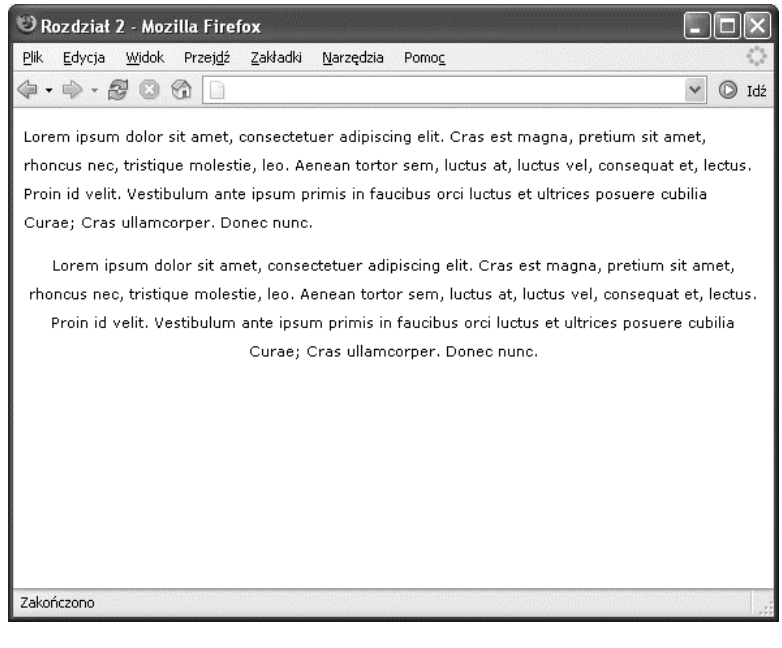
Listing 2.22. *Plik wysrodkowanie.html (fragment)*

```
<p class="wysrodkowany">Lorem ipsum dolor sit amet, consectetuer
   adipiscing elit. Vivamus mollis. Etiam aliquet. Ut ultrices
   justo ut arcu. Proin a purus. Fusce pharetra ultrices nibh.
   Nam erat lectus, dapibus id, congue vel, cursus a, tellus. Sed
   turpis ante, condimentum at, accumsan eget, pulvinar vitae,
   nibh.</p>
```

Listing 2.23. *Plik wysrodkowanie.css (fragment)*

```
.wysrodkowany {
   text-align: center;
}
```

Rezultat widoczny jest na rysunku 2.20.



Rysunek 2.20. *Wyśrodkowanie tekstu za pomocą właściwości text-align*

Jak za pomocą CSS zamienić wszystkie litery w tekście na wielkie

Rozwiązanie

Aby zamienić wszystkie litery w tekście na wielkie lub przeprowadzić inne przekształcenia tekstu, należy skorzystać z właściwości `text-transform` (listing 2.24 i 2.25).

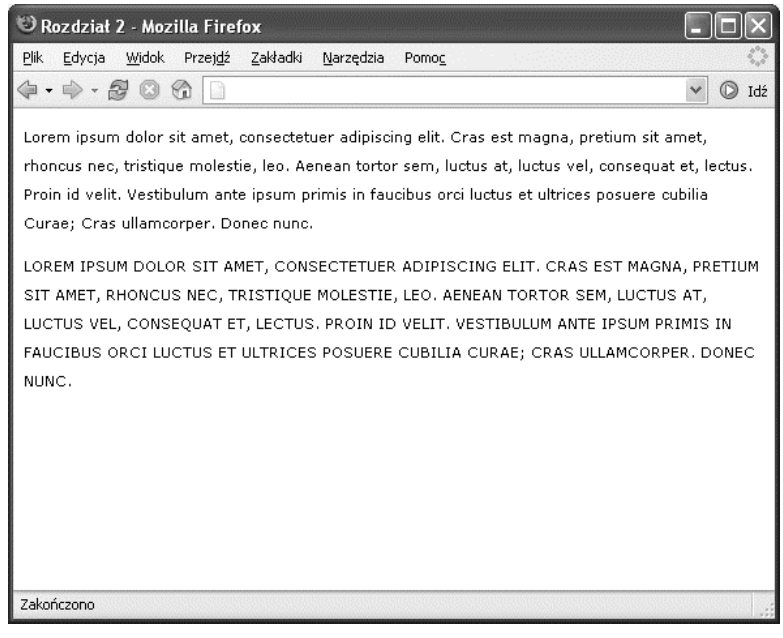
Listing 2.24. *Plik wielkie_litery.html (fragment)*

```
<p class="przeksztalcenie">Lorem ipsum dolor sit amet, consectetuer
   adipiscing elit. Vivamus mollis. Etiam aliquet. Ut ultrices
   justo ut arcu. Proin a purus. Fusce pharetra ultrices nibh. Nam
   erat lectus, dapibus id, congue vel, cursus a, tellus. Sed
   turpis ante, condimentum at, accumsan eget, pulvinar vitae,
   nibh.</p>
```

Listing 2.25. *Plik wielkie_litery.css (fragment)*

```
.przeksztalcenie {
   text-transform: uppercase;
}
```

Zwróć uwagę na wielkie litery w tekście na rysunku 2.21.



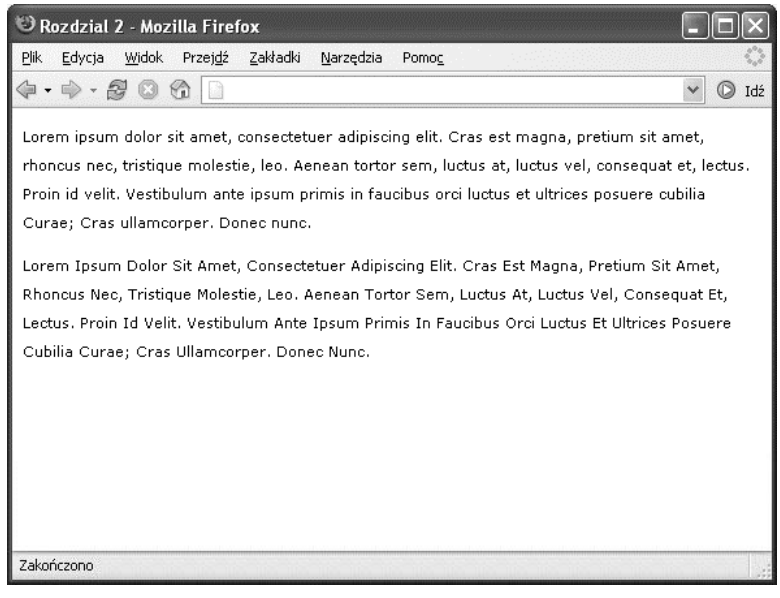
Rysunek 2.21. *Efekt działania właściwości uppercase*

Analiza

Właściwość `text-transform` pozwala przeprowadzić wiele innych przekształceń tekstu. Na przykład wartość `capitalize` spowoduje, że każdy wyraz będzie rozpoczynał się wielką literą (listing 2.26 i rysunek 2.22).

Listing 2.26. *Plik poczatek_wielka.css (frament)*

```
.przeksztalcenie {
  text-transform: capitalize;
}
```



Rysunek 2.22. *Modyfikacja wyglądu tekstu za pomocą właściwości text-transform*

Inne dopuszczalne wartości właściwości `text-transform` to:

- `lowercase`,
- `none` (wartość domyślna).

Jak zmienić lub usunąć znaki punktora na liście elementów

Rozwiązanie

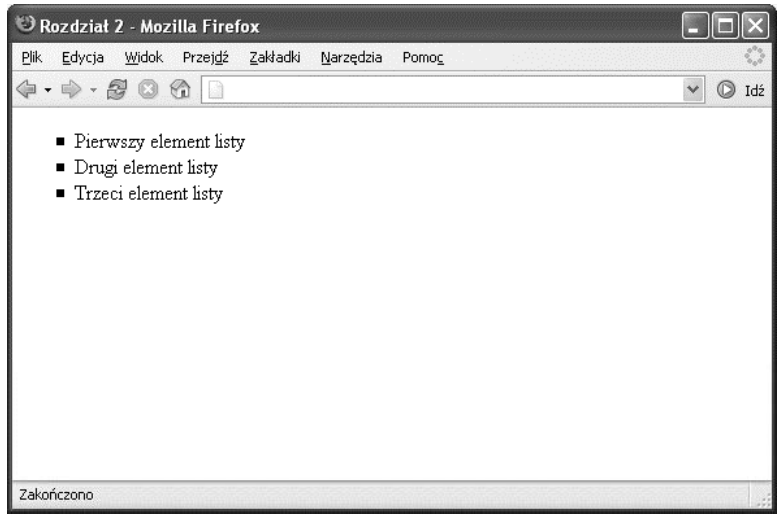
Wygląd punktorów listy elementów można zmodyfikować za pomocą właściwości `list-style-type` (listing 2.27).

Listing 2.27. *Plik punktory.html (fragment)*

```
<ul>
   <li>Pierwszy element listy</li>
   <li>Drugi element listy</li>
   <li>Trzeci element listy</li>
</ul>
```

Aby użyć punktorów kwadratowych (rysunek 2.23), należy właściwości `list-style-type` przypisać wartość `square` (listing 2.28).

Rysunek 2.23. *Lista elementów wyświetlona z wykorzystaniem punktorów kwadratowych*



Listing 2.28. *Plik punktory.css*

```
ul {
   list-style-type: square;
}
```

Analiza

Oto pozostałe dopuszczalne wartości właściwości `list-style-type`:

- `disc`,
- `circle`,
- `decimal-leading-zero`,
- `decimal`,
- `lower-roman`,
- `upper-roman`,
- `lower-greek`,

- `lower-alpha`,
- `lower-latin`,
- `upper-alpha`,
- `upper-latin`,
- `Hebrew`,
- `Armenian`,
- `Georgian`,
- `none`.

Nie wszystkie wymienione wartości są obsługiwane przez każdą przeglądarkę; jeżeli dana wartość nie zostanie rozpoznana, zastosowany zostanie punktor domyślny. Aby sprawdzić, które rodzaje punktorów są obsługiwane przez Twoją przeglądarkę, możesz wejść na stronę *http://www.meyerweb.com/eric/css/tests/css2/sec12-06-02a.htm*. Ustawienie właściwości `list-style-type` jako `none` (listing 2.29) spowoduje, że nie będą wyświetlane jakiekolwiek punktory, jednak zawartość listy pozostanie wcięta w stosunku do reszty tekstu. Taką sytuację przedstawia rysunek 2.24.

Listing 2.29. *Plik punktory2.css*

```
ul {
   list-style-type: none;
}
```

Rysunek 2.24. *Lista bez punktorów*

Jak zdefiniować punktor użytkownika

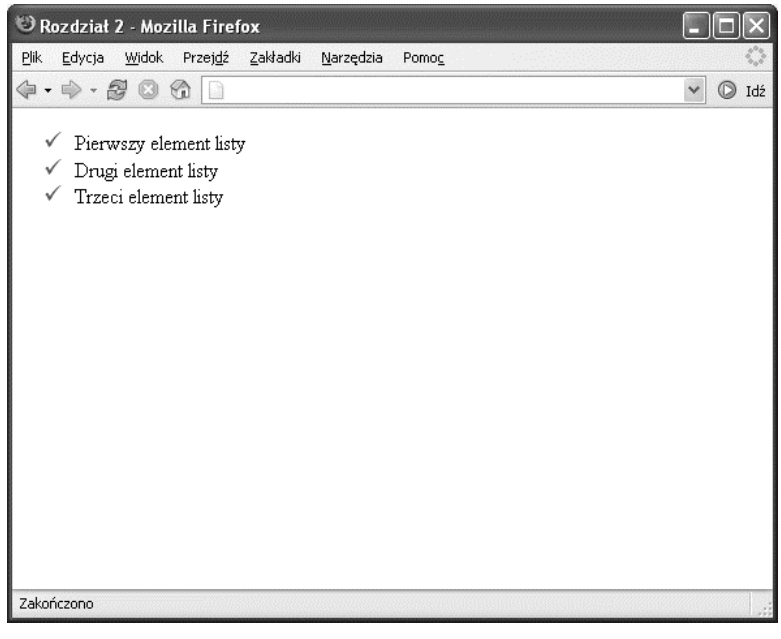
Rozwiązanie

Aby zastosować własny obrazek jako punktor listy numerowanej, należy zamiast właściwości `list-style-type` użyć właściwości `list-style-image`, która pozwala podać adres URL zawierający ścieżkę i nazwę pliku graficznego. Zostało to pokazane na listingu 2.30.

Listing 2.30. *Plik wlasny_punktor.css*

```
ul {
   list-style-image: url(bullet.gif);
}
```

Przykład zastosowania własnego punktora przedstawia rysunek 2.25.



Rysunek 2.25. *Zastosowanie własnego obrazka jako punktora listy elementów*

Wiele punktorów na jednej liście

Właściwość `list-style-image` stosuje się do modyfikacji znaczników elementów listy `<li>`, jednak można użyć jej także w odniesieniu do listy jako całości i wtedy poszczególne elementy listy odziedziczą wartość właściwości. Korzystając z pierwszego rozwiązania, można przypisać każdemu elementowi listy inną wartość właściwości `list-style-image` i uzyskać przez to inny wygląd każdego z punktorów.

Jak usunąć wcięty lewy margines elementów listy

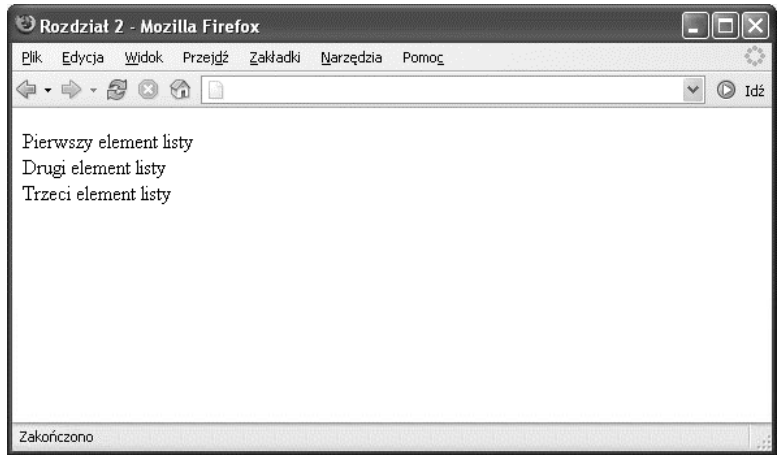
Jeżeli ustawisz wartość właściwości `list-style-type` na `none`, być może zechcesz również zrezygnować z lewego marginesu lub go zmniejszyć (domyślnie to do niego wyrównywane są elementy listy).

Rozwiązanie

Aby całkowicie usunąć wcięcie i wyrównać elementy listy z pozostałą zawartością strony (tak jak widać to na rysunku 2.26), należy zastosować kod pokazany na listingu 2.31.

Listing 2.31. *Plik lista_bez_wciecia.css*

```
ul {
   list-style-type: none;
   padding-left: 0;
   margin-left: 0;
}
```



Rysunek 2.26. *Lista bez wcięć i punktorów*

Analiza

Możesz również dostosować wielkość wcięcia do własnych wymagań. Aby wciąć listę o kilka pikseli, spróbuj zdefiniować następujący styl:

```
ul {
   list-style-type: none;
   padding-left: 5px;
   margin-left: 0;
}
```

Jak wyświetlić elementy listy w poziomie

Domyślnie elementy listy wyświetlane są jako elementy blokowe, czyli każdy w oddzielnej linii. Czasem jednak zachodzi konieczność umieszczenia elementów listy jeden po drugim.

Rozwiązanie

Aby lista została wyświetlona w poziomie, należy zdefiniować właściwość `display` znacznika `<li>` jako `inline` (listing 2.32 i 2.33).

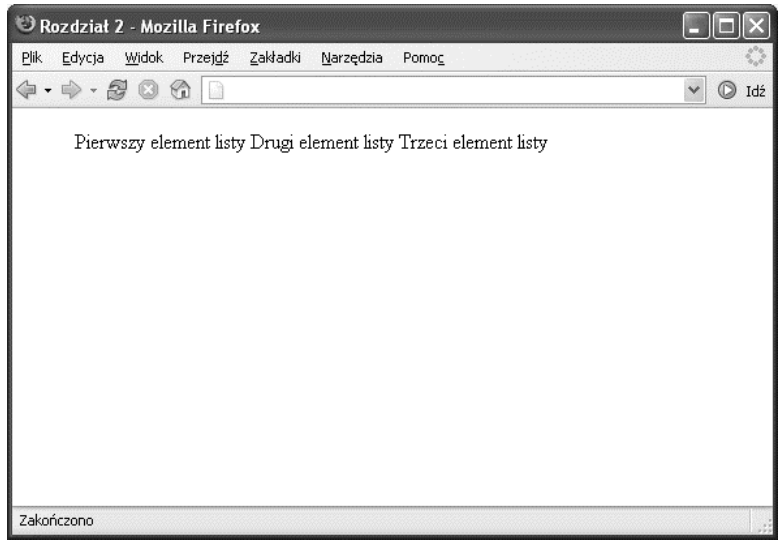
Listing 2.32. *Plik lista_poziomo.html (fragment)*

```
<ul class="poziomo">
   <li>Pierwszy element listy</li>
   <li>Drugi element listy</li>
   <li>Trzeci element listy</li>
</ul>
```

Listing 2.33. *Plik lista_poziomo.css*

```
ul.poziomo li {
   display: inline;
}
```

Wynik działania powyższego kodu został pokazany na rysunku 2.27.



Rysunek 2.27. *Elementy listy mogą być wyświetlane w poziomie*

Jak dodać komentarz do kodu w pliku CSS

Można i powinno się umieszczać komentarze do kodu CSS. Jeżeli plik CSS jest bardzo prosty i posiada tylko kilka reguł dotyczących na przykład formatowania tekstu, komentowanie może być zbędne. Jednak jeśli zaczniesz tworzyć duże pliki CSS z licznymi definicjami stylów i rozbudowanym kodem, wówczas komentarze bardzo się przydadzą. Po pewnym czasie na pewno zapomnisz o szczegółach swoich pomysłów i bez komentarzy możesz stracić mnóstwo czasu na szukanie określonej klasy czy zastanawianiu się, co robi dany fragment kodu lub gdzie dokładnie ma zastosowanie.

Rozwiązanie

CSS umożliwia stosowanie komentarzy zajmujących więcej niż jedną linię, podobnie jak w JavaScript. Aby oznaczyć wybrany fragment kodu jako komentarz, należy użyć następującej konstrukcji:

```
/*
 ...
*/
```

Absolutne minimum to umieszczenie komentarza na początku każdego arkusza stylów z opisem zawartości arkusza.

```
/* To jest domyślny arkusz stylów dla całego tekstu witryny */
```

Jak usunąć marginesy bez dodawania atrybutów do znacznika <body>

Aby bez stosowania CSS usunąć domyślny odstęp między dokumentem a oknem przeglądarki, należy dodać odpowiednie atrybuty do znacznika `<body>`:

```
<body topmargin="0" leftmargin="0" marginheight="0" marginwidth="0">
```

Rozwiązanie

W specyfikacji HTML odradza się stosowanie powyższych atrybutów. Powinno się je zastępować kodem CSS zdefiniowanym dla znacznika `<body>`:

```
body {
   margin: 0;
   padding: 0;
}
```

Przeglądarka Opera robi to po swojemu

W niektórych wersjach Opery marginesy i odstępy między wierszami są definiowane w elemencie <html> zamiast w znaczniku <body>. Z tego powodu, aby zachować kompatybilność, należy użyć nieznacznie zmodyfikowanego kodu:

```
html, body {
   margin: 0;
   padding: 0;
}
```

Podsumowanie

W rozdziale zostały zawarte informacje na temat podstawowych technik CSS. Przydadzą się osobom, które wcześniej nie miały kontaktu z kaskadowymi arkuszami stylów. Postawione pytania dotyczyły formatowania tekstu oraz stylów tekstowych. Stosując przedstawione techniki, można uzyskać bardzo atrakcyjne efekty, które będą jednak niedostępne dla osób korzystających z przeglądarek nieobsługujących CSS.