

IDŹ DO

PRZYKŁADOWY ROZDZIAŁ



SPIS TREŚCI

KATALOG KSIĄŻEK

KATALOG ONLINE

ZAMÓW DRUKOWANY KATALOG

TWÓJ KOSZYK

DODAJ DO KOSZYKA

CENNIK I INFORMACJE

ZAMÓW INFORMACJE
O NOWOŚCIACH

ZAMÓW CENNIK

CZYTELNIA

FRAGMENTY KSIĄŻEK ONLINE

Po prostu HTML 4. Wydanie III

Autor: Elizabeth Castro

Tłumaczenie: Piotr Rajca

ISBN: 83-7361-039-1

Tytuł oryginału: [HTML for the World Wide Web
with XHTML and CSS VQG](#)

Format: B5, stron: 490



Wśród książek poświęconych językowi HTML, „Po prostu HTML 4. Wydanie III” zajmuje szczególną pozycję. To prawdziwy bestseller i najpopularniejsza w Polsce książka o HTML-u, która doczekała się już trzech edycji. Najnowsze wydanie przynosi najświeższe informacje na temat języka XHTML, nowego standardu tworzenia stron internetowych. Nowością są liczne porady niezbędne webmasterom opracowującym strony internetowe dedykowane urządzeniom przenośnym (palmtopom, telefonom komórkowym...).

Szczególną zaletą książki jest przejrzysty układ i prosty język, którym została napisana. Zamiast długich objaśnień znajdziesz w niej krótkie, trafne i zwięzłe informacje. Towarzyszące im liczne ilustracje sprawiają, że przyswajanie przedstawionej w niej wiedzy jest szybkie i łatwe.

Dzięki książce:

- Stworzysz własne strony wykorzystując HTML i XHTML
- Nauczysz się stosować CSS (kaskadowe arkusze stylów)
- Dowiesz się, jak tworzyć „elastyczne układy” stron, dostosowujące się do wielkości ekranu użytkownika
- Uczynisz strony atrakcyjniejszymi umieszczając klipy wideo, pliki dźwiękowe w formacie MP3 oraz inne multimedia bezpośrednio na stronach WWW
- Korzystając z WML-a stworzysz strony przeznaczone dla urządzeń przenośnych
- Skorzystasz z bogatego doświadczenia autorki książki

Z poprzednich wydań tej książki tysiące ludzi nauczyło się tworzyć strony WWW. Być może odwiedzałeś ich witryny w Internecie. Teraz i Ty masz szansę poznania tajników WWW i zaprosić wszystkich ma samodzielnie stworzoną stronę.



Spis treści

	Wstęp	13
	Internet, WWW oraz HTML.....	14
	Otwarty, jednak nie identyczny	15
	Wojny przeglądarek	16
	Dążenie do wprowadzania standardów.....	17
	Rzeczywistość	20
	Czego należy używać?	21
	Kilka słów o niniejszej książce.....	23
	Witryna WWW Autorki	24
Rozdział 1.	Elementy tworzące strony WWW	25
	Znaczniki: elementy, atrybuty oraz wartości.....	26
	Tekstowa zawartość stron WWW.....	30
	Łącza, obrazy oraz inna zawartość nie będąca tekstem	31
	Nazwy plików	32
	Adresy URL.....	33
	HTML a XHTML.....	36
	Wersje, rodzaje oraz DOCTYPE.....	38
	Domyślny sposób wyświetlania (X)HTML	40
	Dodawanie stylów do stron WWW.....	41
	Kaskada: kiedy style kolidują ze sobą.....	42
	Wartości właściwości.....	44
Rozdział 2.	Praca z dokumentami (X)HTML	47
	Projektowanie witryny	48
	Tworzenie nowej strony WWW	49
	Zapisywanie stron WWW.....	50
	Kilka słów o programie Microsoft Word i stronach WWW.....	52
	Określanie strony domyślnej („domowej”).....	53
	Edycja stron WWW.....	54
	Organizowanie plików.....	55
	Wyświetlanie stron w przeglądarce.....	56
	Czerpanie inspiracji od innych	57
Rozdział 3.	Podstawowa struktura dokumentów (X)HTML	59
	Rozpoczynanie strony	60
	Tworzenie podstawowej struktury kodu.....	62
	Określanie sposobu kodowania	63
	Nadawanie tytułu.....	64
	Tworzenie nagłówków sekcji.....	65
	Rozpoczynanie nowego akapitu	66

	Nazywanie elementów.....	67
	Podział strony na dział.....	68
	Tworzenie obszarów wewnątrzwierszowych.....	69
	Tworzenie nowych wierszy.....	70
	Dodawanie komentarzy.....	71
	Nadawanie nazw elementom stron.....	72
Rozdział 4.	Podstawy formatowania tekstu	73
	Tworzenie tekstu pogrubionego oraz kursywy.....	74
	Zmiana wielkości tekstu.....	75
	Stosowanie czcionki o stałej szerokości znaków.....	76
	Stosowanie tekstu preformatowanego.....	77
	Cytaty.....	78
	Tworzenie indeksów dolnych i górnych.....	80
	Oznaczanie zmodyfikowanego tekstu.....	81
	Wyjaśnianie skrótów.....	82
Rozdział 5.	Tworzenie obrazów na potrzeby stron WWW	83
	O obrazach tworzonych na potrzeby stron WWW.....	84
	Zdobywanie obrazów.....	89
	Polecenie Zapisz dla Webu.....	90
	Zmniejszanie wymiarów obrazu.....	92
	Tworzenie przezroczystości.....	93
	Zapisywanie obrazu z przezroczystymi obszarami.....	94
	Symulacja przezroczystości.....	95
	Wykorzystanie (przeważnie) bezpiecznych kolorów.....	96
	Redukcja ilości kolorów.....	98
	Progresywne wyświetlanie obrazu.....	99
	Rozmywanie obrazków JPEG dla ułatwienia kompresji.....	100
	Tworzenie animowanych obrazów GIF.....	101
Rozdział 6.	Wykorzystanie obrazów	103
	Wstawianie ilustracji na stronę.....	104
	Tekst zastępczy.....	105
	Określanie wymiarów obrazu w celu jego szybszego wyświetlenia.....	106
	Skalowanie ilustracji.....	108
	Łączenie ikon z obrazami.....	109
	Otoczanie obrazów tekstem.....	110
	Zakończenie otaczania tekstem.....	112
	Zwiększanie odstępu wokół obrazów.....	113
	Wyrównywanie obrazków.....	114
	Poziome linie.....	115
Rozdział 7.	Łącza	117
	Tworzenie łączy do innych stron.....	118
	Odnosińki.....	120
	Łącza wykorzystujące odnośniki.....	121
	Tworzenie łączy do wybranego okna.....	122

	Określenie domyślnego okna	123
	Tworzenie innych rodzajów łączy	124
	Definiowanie klawiszy skrótów dla łączy	126
	Określenie kolejności łączy dla klawisza TAB	127
	Przyciski nawigacyjne	128
	Podział obrazka na obszary przypisane do różnych łączy	129
	Tworzenie map odnośników obsługiwanych po stronie klienta	130
	Korzystanie z mapy interpretowanej na serwerze	132
Rozdział 8.	Tworzenie stylów	133
	Tworzenie reguły stylu	134
	Tworzenie selektorów	135
	Wybór elementów na podstawie nazwy	136
	Wybieranie elementu na podstawie klasy lub identyfikatora	137
	Wybieranie elementów na podstawie kontekstu	138
	Wybieranie łączy na podstawie ich stanu	141
	Wybieranie fragmentów elementu	142
	Wybieranie elementów na podstawie atrybutów	144
	Definiowanie grup elementów	145
	Łączenie selektorów	146
Rozdział 9.	Stosowanie stylów	147
	Tworzenie zewnętrznego arkusza stylów	148
	Dołączanie zewnętrznego arkusza stylów	149
	Udostępnianie alternatywnych arkuszy stylów	150
	Tworzenie wewnętrznego arkusza stylów	151
	Importowanie zewnętrznych arkuszy stylów	152
	Lokalne stosowanie stylów	153
	Znaczenie położenia	154
	Dodawanie komentarzy do arkuszy stylów	155
Rozdział 10.	Formatowanie przy wykorzystaniu stylów	157
	Wybór czcionki	158
	Osadzanie czcionek w dokumencie	159
	Tworzenie czcionki pochylej (kursywy)	160
	Pogrubianie czcionki	161
	Określanie wielkości czcionki	162
	Określanie wysokości linii	164
	Jednoczesne określanie wszystkich parametrów czcionki	165
	Definiowanie koloru tekstu	166
	Zmiana koloru tła tekstu	167
	Kontrola odstępów pomiędzy wyrazami i literami	168
	Dodawanie wcięć akapitowych	169
	Parametry odstępów	170
	Wyrównywanie tekstu	171
	Zmiana wielkości liter	172
	Wykorzystanie kapitalików	173
	Dekorowanie tekstu	174

Rozdział 11.	Określanie układu strony za pomocą stylów	175
	Określanie struktury strony.....	176
	Model prostokątów	177
	Wyświetlanie i ukrywanie elementów.....	178
	Bezwzględne rozmieszczanie elementów.....	179
	Określanie stałego położenia elementu w oknie przeglądarki	180
	Przesuwanie elementów względem ich naturalnego położenia.....	181
	Modyfikowanie tła.....	182
	Zmiana koloru	184
	Modyfikacja wskaźnika myszy.....	185
	Tworzenie obramowań.....	186
	Dodawanie wypełnienia wokół elementu	188
	Określanie marginesów wokół elementu	189
	Określanie wysokości i szerokości elementu.....	190
	Pozycjonowanie elementów w trzecim wymiarze	192
	Określanie sposobu wyświetlania zawartości elementu.....	193
	Otaczanie elementów tekstem	194
	Kontrola sposobu otaczania elementów.....	195
	Wyrównywanie elementów w pionie	196
Rozdział 12.	Arkusze stylów przeznaczone do drukowania stron	197
	Stosowanie arkuszy stylów przeznaczonych dla konkretnych rodzajów mediów	198
	Czym różnią się style przeznaczone do drukowania	199
	Kontrola dzielenia dokumentów na strony.....	200
	Inne właściwości CSS, charakterystyczne dla sporządzania wydruków	201
Rozdział 13.	Listy	203
	Tworzenie list wypunktowanych i uporządkowanych.....	204
	Określanie kształtu znaczników (punktów).....	206
	Określanie początkowej wartości numeracji punktów.....	207
	Stosowanie niestandardowych znaczników.....	208
	Określanie miejsca wyświetlania znaczników	209
	Określanie wszystkich właściwości listy w jednym miejscu.....	210
	Tworzenie list definicji.....	211
	Określanie wyglądu list zagnieżdżonych.....	212
Rozdział 14.	Tabele	215
	Projektowanie układu strony.....	216
	Tworzenie prostej tabeli	217
	Dodawanie krawędzi tabeli.....	218
	Określanie szerokości tabel.....	220
	Wyrównywanie tabeli do środka strony	222
	Otaczanie tabeli tekstem	223
	Łączenie tabel.....	224
	Wyrównywanie zawartości komórek	226
	Zmiana koloru tła.....	228

Kontrola odstępów pomiędzy i wewnątrz komórek	230
Łączenie komórek leżących w sąsiednich kolumnach	232
Łączenie komórek w sąsiednich wierszach	233
Podział tabeli na grupy kolumn	234
Podział tabeli na poziome sekcje	236
Wybór linii do wyświetlania	237
Kontrola łamania wierszy w komórce	239
Przyspieszenie wyświetlania tabeli	240

Rozdział 15. **Ramki** **241**

Tworzenie prostego układu ramek	242
Ramki w kolumnach	244
Tworzenie ramek w wierszach i kolumnach	245
Kombinowany układ ramek	246
Ramki wpisane	248
Określanie wielkości marginesów ramki	249
Wyświetlanie lub chowanie pasków przewijania	250
Wybieranie koloru krawędzi	251
Modyfikacja grubości krawędzi	252
Uniemożliwienie użytkownikowi zmiany rozmiarów ramki	254
Wyświetlanie łączy w konkretnych ramkach	255
Określanie miejsca docelowego dla łącza	256
Zmiana domyślnego miejsca docelowego	257
Zagnieżdżanie układów ramek	258
Tworzenie zamiennika ramek	259
Osadzanie zawartości przy użyciu obiektów	260
Zapewnianie większej dostępności ramek	262

Rozdział 16. **Formularze** **263**

Skrypty CGI	264
Zdobywanie skryptów	266
Wykorzystanie skryptów dołączonych do tej książki	267
Przygotowanie skryptu do użycia	268
Tworzenie formularza	269
Przesyłanie danych pocztą elektroniczną	270
Wykorzystanie serwisów obsługujących formularze	271
Tworzenie pól tekstowych	272
Tworzenie pól hasła	273
Tworzenie przycisków opcji	274
Tworzenie pól wyboru	275
Menu	276
Obszary tekstowe	278
Umożliwienie użytkownikom przysyłania plików	279
Kilka uwag o polach ukrytych	280
Dodawanie pól ukrytych do formularzy	281
Tworzenie przycisku wysyłającego	282
Czyszczenie zawartości formularza	284
Aktywne obrazki	286
Organizacja elementów formularzy	287

Formalne nadanie etykiety elementowi formularza.....	288
Określenie kolejności klawisza TAB w formularzach.....	289
Definiowanie klawiszy skrótów.....	290
Dezaktywacja elementów formularza.....	291
Uniemożliwienie modyfikacji elementów.....	292

Rozdział 17. **Multimedia** **293**

Aplikacje pomocnicze i moduły dodatkowe (plug-ins).....	294
Zdobywanie odtwarzaczy dla użytkowników.....	296
Pobierania plików multimedialnych.....	297
Osadzanie w stronach filmów QuickTime.....	298
Skalowanie filmów QuickTime.....	300
Odtwarzanie filmów QuickTime w pętli.....	301
Umieszczanie dźwięków QuickTime na stronach WWW.....	302
Ukrywanie dźwięków QuickTime.....	303
Odtwarzanie plików za pomocą programu Windows Media Player.....	304
Dołączanie apletów.....	306
Osadzanie innych plików multimedialnych.....	307
Dołączanie plików multimedialnych.....	308
Tworzenie automatycznego pokazu slajdów.....	309
Tworzenie szyldów.....	310
Dźwięk odtwarzany w tle.....	311

Rozdział 18. **Skrypty** **313**

Wstawianie „automatycznego” skryptu.....	314
Wywołanie zewnętrznego skryptu automatycznego.....	315
Wyzwalanie skryptu.....	316
Tworzenie przycisku, który wykonuje skrypt.....	318
Dodawanie informacji zastępczych.....	319
Ukrywanie skryptu przed starszymi przeglądarkami.....	320
Ukrywanie skryptów przed analizatorami składni XML.....	321
Definiowanie domyślnego języka skryptowego.....	322

Rozdział 19. **Podstawy JavaScriptu** **323**

Dodawanie aktualnej daty i godziny.....	324
Zmiana etykiety stanu łącza.....	325
Zmiana zawartości kilku ramek przy użyciu jednego łącza.....	326
Wyświetlanie stron w odpowiednich ramkach.....	327
Podmienianie obrazków po wskazaniu ich myszką.....	328
Ładowanie obrazków do pamięci podręcznej.....	330
Określanie wielkości nowego okna przeglądarki.....	331

Rozdział 20. **Symbole oraz inne znaki nie należące do alfabetu angielskiego** **333**

Kilka słów o sposobach kodowania.....	334
Zapisywanie stron przy użyciu odpowiedniego sposobu kodowania.....	336
Edytowanie stron przy wykorzystaniu poprawnego sposobu kodowania.....	337

Deklarowanie sposobu kodowania strony.....	338
Dodawanie znaków nie należących do danego sposobu kodowania.....	340
Określanie języka strony	342
Rozdział 21. Stare sposoby formatowania	343
Określanie domyślnej postaci tekstu	344
Formatowanie fragmentów tekstu.....	346
Inny sposób określania domyślnych kolorów.....	348
Zmienianie koloru łączy	349
Przekreślenia i podkreślenia tekstu	350
Tekst migający	351
Rozdział 22. Określanie układu strony — stary sposób	353
Kolor tła.....	354
Stosowanie obrazów tła.....	355
Wyśrodkowanie elementów na stronie	356
Określenie marginesów	357
Zapobieganie dzieleniu wierszy.....	358
Warunkowe łamanie wierszy	359
Określanie odstępu pomiędzy akapitami	360
Tworzenie wcięć.....	361
Tworzenie wcięć za pomocą list.....	362
Wstawianie pustych prostokątów	363
Wykorzystanie prostokątów pikselowych.....	364
Wykorzystanie kolumn.....	365
Rozmieszczanie elementów na warstwach.....	366
Rozdział 23. WML: strony WWW dla urządzeń przenośnych	369
Przygotowywanie serwera	370
Początek tworzenia strony WML	371
Tworzenie karty	372
Tworzenie podstawowej zawartości.....	373
Dołączanie obrazów.....	374
Tworzenie tabel.....	375
Tworzenie odnośników	376
Programowanie przycisków	378
Tworzenie zadań warunkowych	380
Planowanie zadań	381
Nawiązywanie połączenia.....	382
Tworzenie i stosowanie zmiennych.....	383
Tworzenie pól tekstowych.....	384
Tworzenie list.....	386
Przetwarzanie informacji podawanych przez użytkowników	388
Tworzenie elementów na wielu stronach	390
Ograniczanie dostępu do talii	391
Testowanie stron WML.....	392

Rozdział 24.	Testowanie stron WWW	393
	Sprawdzanie poprawności kodu	394
	Sprawdzanie prostych błędów: HTML.....	395
	Sprawdzanie prostych błędów: XHTML.....	396
	Sprawdzanie prostych błędów: CSS.....	397
	Testowanie stron	398
	Gdy przeglądarka wyświetla kod.....	400
	Obrazki nie są wyświetlane.....	401
	Wspaniałe w jednej przeglądarce, a brzydkie w innych.....	402
	Gdy strona nie jest wyświetlana w Netscape 4.....	403
	Wciąż nie działa?.....	404
Rozdział 25.	Publikowanie stron w sieci WWW	405
	Jak znaleźć serwer dla swoich stron.....	406
	Gdzie w Polsce opublikować strony WWW?.....	407
	Onet.pl — Republika WWW	411
	Rejestracja nazwy domeny	419
	Przesyłanie plików na serwer.....	420
Rozdział 26.	Zdobywanie użytkowników	423
	O słowach kluczowych.....	424
	Jawne podawanie słów kluczowych.....	425
	Opis strony.....	426
	Zarządzanie innymi informacjami o stronie	427
	Jak uniknąć odwiedzin	428
	Zapobieganie archiwizacji strony	429
	Tworzenie strony z adresami.....	430
	Dodawanie witryny do wyszukiwarki	432
	Jak zapewnić wysoką pozycję strony w wynikach wyszukiwania	433
	Pisanie stron łatwych do indeksowania	434
	Inne sposoby reklamowania witryny	435
Dodatek A	Elementy i atrybuty (X)HTML	437
Dodatek B	Właściwości i wartości CSS	447
Dodatek C	Zdarzenia wbudowane	455
Dodatek D	Symbole i znaki (X)HTML	457
Dodatek E	Wartości szesnastkowe	467
Dodatek F	Narzędzia (X)HTML	469
	Skorowidz	473

Określanie układu strony za pomocą stylów

11



Rysunek 11.1. Układ tej strony został określony przy wykorzystaniu CSS. Został on szczegółowo i dokładnie wyjaśniony w dalszej części niniejszego rozdziału

Wykorzystanie CSS do określania układu strony ma kilka zalet w porównaniu z innymi metodami, takimi jak na przykład tabele (które opisałam w rozdziale 14.). Przede wszystkim CSS doskonale nadają się do tworzenia *elastycznych układów*, które poszerzają się lub zwężają w zależności od wielkości okna przeglądarki. Poza tym oddzielenie zawartości strony od instrukcji określających jej wygląd daje możliwość bardzo łatwego wykorzystania tego samego układu w całej witrynie. Dzięki temu, modyfikując jedynie plik CSS, można także bardzo łatwo i szybko zmienić wygląd całej witryny. Co więcej, połączenie CSS z (X)HTML pozwala zmniejszyć rozmiar plików, co z kolei oznacza krótszy czas oczekiwania na wyświetlenie witryny. W końcu, ponieważ CSS i (X)HTML są aktualnie używanymi standardami, można mieć gwarancję, że strony utworzone zgodnie z ich założeniami będą poprawnie wyświetlane w przyszłych wersjach przeglądarek (poza tym stosowanie tych standardów stanie się koniecznością dla profesjonalnych projektantów stron).

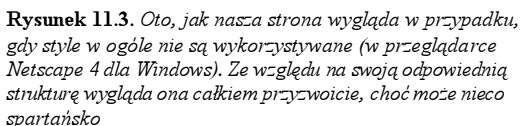
Podstawową wadą kaskadowych arkuszy stylów, w szczególności w odniesieniu do określania układu stron, jest fakt, iż starsze przeglądarki ich nie obsługują lub obsługują je w sposób niewłaściwy. W szczególności przeglądarka Netscape 4.x cechuje się wyjątkowo złą obsługą możliwości CSS związanych z określaniem układu strony. Nawet w najnowszych wersjach przeglądarek nie wszystkie możliwości CSS są obsługiwane poprawnie. Często też twórcy różnych przeglądarek mają odmienne opinie odnośnie tego, które rozwiązania należy uznać za poprawne. Istnieją jednak sposoby na umożliwienie oglądania stron o układach tworzonych za pomocą stylów osobom korzystającym ze starszych przeglądarek oraz na ominięcie różnic w działaniu nowych wersji przeglądarek; informacje na ten temat zamieściłam w rozdziale 9., „Stosowanie stylów”.

Przykłady użyte w tym rozdziale — prześlicznie zaprojektowane przez Erica Costello (<http://www.glish.com/>) — wyglądają doskonale w nowoczesnych przeglądarkach, ale także w ich starszych wersjach nie tracą wiele na swej atrakcyjności (rysunek 11.1).

Najważniejszym aspektem CSS jest rozdzielenie zawartości strony i reguł formatujących oraz określających styl. W ten sposób pozbawiamy się ścisłych dyrektyw definiujących wygląd strony i uzyskujemy elastyczność, gwarantującą poprawną prezentację strony w różnych przeglądarkach, systemach operacyjnych oraz urządzeniach wyjściowych (nawet na wydrukach). Prawdopodobnie najważniejszą cechą stron, których układ ma być określany za pomocą arkuszy stylów, jest ich wewnętrzna struktura. Przeniesiła, logiczną strukturę będzie znacznie łatwiej dostosować do więcej niż jednego rodzaju urządzeń wyjściowych.

1. Utwórz logiczne sekcje dokumentu w formie elementów `div`. Na naszej przykładowej stronie można wyróżnić dział *nawigacyjny* (ang. *navigation*), w którym zostaną umieszczone połączenia z innymi stronami, dział *bg* — zawierający obraz tła działu nawigacyjnego oraz dział *treści* (ang. *content*) — zawierający podstawową treść strony, czyli tekst i obrazki. Strona zawiera także dział *kalendaryzacyjny*, będący ruchomym paskiem bocznym, służącym do wyświetlania przyszłych zdarzeń.
2. Rozmieść elementy `div` w kolejności, która ma sens nawet wtedy, gdy arkusze stylów nie są obsługiwane. Na przykład na samym początku strony umieść tytuł, za nim spis treści, a na końcu właściwą treść strony. Dzięki temu wciąż będziesz mógł dowolnie rozmieszczać elementy, wykorzystując CSS, a strona zawsze będzie mieć sensowny wygląd — nawet jeśli style nie będą obsługiwane (oczywiście kolejność rozmieszczenia elementów zależy od sytuacji, docelowej grupy odbiorców oraz wielu innych czynników. Pamiętaj, aby cały czas zastanawiać się nad sposobami zapewnienia dostępności strony, nawet w przypadkach, gdy CSS nie będą stosowane).
3. W spójny sposób używaj elementów tworzących nagłówki (`h1`, `h2` i tak dalej), aby określić znaczenie informacji umieszczonych na stronie.
4. Stosuj komentarze, aby wskazać różne fragmenty strony oraz opisać ich zawartość.

Rysunek 11.2. Oto dokument, którego będę używać w tym rozdziale. Zwróć uwagę, że składa się on z trzech podstawowych działów — *navigation* (części nawigacyjne), *bg* oraz *content* (treść). Pełny kod strony można znaleźć w przykładach dołączonych do książki.





Rysunek 11.4. Prostokąt każdego elementu posiada cztery istotne cechy określające jego wielkość (podane tutaj w kolejności od środka obszaru elementu): obszar zawartości, wypełnienie oraz margines. Każdą z tych właściwości (a nawet wybrane cechy każdej z nich) można określać niezależnie od pozostałych



Rysunek 11.5. Każdy element posiada własny prostokąt. W tym przykładzie elementami blokowymi są działy strony, nagłówki różnego stopnia, akapity, jak również połączenia nawigacyjne („Strona główna”, „Barcelona”, i tak dalej), które, choć normalnie są elementami wewnętrznymi, w tym przykładzie zostały zmienione na elementy blokowe. Zwróć uwagę, że elementy wewnętrznymi, takie jak em (na przykład: „budowanie stajni” oraz „Castellers”), nie generują nowych wierszy

Model prostokątów

W CSS strona jest traktowana w taki sposób, jak gdyby każdy z umieszczonych na niej elementów był zawarty w niewidocznym prostokącie. Prostokąt ten składa się z obszaru zawierającego treść elementu, obszaru otaczającego obszar treści (nazywanego *wypełnieniem*), krawędzi wypełnienia, czyli ramek, oraz niewidzialnego obszaru umieszczonego poza krawędziami (marginesów).

Wykorzystując CSS, można określić położenie prostokąta każdego elementu strony, co daje nam znaczącą kontrolę nad jej układem.

Zgodnie z tym, co podałem wcześniej (na stronie 28), prostokąt elementu może mieć charakter *pojemnika* (przez co za elementem zostanie utworzony nowy wiersz) lub charakter *wewnętrzny* (i w takim przypadku nowy wiersz za elementem nie będzie tworzony). Właśnie ta cecha określa początkowy układ strony: domyślnie elementy są wyświetlane w kolejności, w jakiej występują w kodzie (X)HTML, przy czym nowe wiersze są tworzone na początku i na końcu prostokątów elementów blokowych.

Istnieją cztery podstawowe sposoby określenia położenia prostokąta elementu: można go pozostawić w naturalnym położeniu (domyślnym, określanym także jako *statyczne*), można usunąć element z naturalnego toku rozmieszczania i określić dokładnie jego położenie w odniesieniu do elementu nadrzędnego (położenie *bezwzględne*) lub całej strony (położenie *ustalone*), można wreszcie też przesunąć element względem jego położenia naturalnego (położenie *względne*). Co więcej, jeśli prostokąty elementów zachodzą na siebie wzajemnie, można określić kolejność, w jakiej powinny być wyświetlane (tak zwany *z-index*).

Po wskazaniu, gdzie należy wyświetlić prostokąt elementu, można zająć się jego wyglądem i określić jego wypełnienie, krawędzie, marginesy, wielkość, wyrównanie, kolor i tak dalej. Wszystkie te cechy zostały opisane w niniejszym rozdziale.

Zwróć uwagę, że niektóre właściwości związane z układem, w szczególności te, których wartości są określane w jednostkach em lub w formie wartości procentowych, są uzależnione od *elementu nadrzędnego*. Pamiętaj, że element nadrzędny to ten, w którym element bieżący jest umieszczony (patrz strona 28).

Wyświetlanie i ukrywanie elementów

Właściwość `display` jest bardzo przydatna do ukrywania lub wyświetlania określonych elementów w zależności od rodzaju przeglądarki użytkownika, jego języka lub dowolnego innego kryterium. Można także zmienić domyślny sposób wyświetlania danego elementu (z blokowego na wewnątrzwierszowy lub na odwrót). Można też sprawić, że wszystkie elementy będą wyświetlane w postaci listy, nawet bez potrzeby stosowania znacznika `li` (to zagadnienie opisałam na stronie 203).

Aby określić, jak wyświetlane będą elementy:

1. Wpisz `display`.
2. Dopisz `none`, aby ukryć dany element, lub wpisz `block`, aby wyświetlić element w postaci blokowej (i tym samym utworzyć nowy akapit), lub wpisz `inline`, aby wyświetlić element w postaci wewnątrzwierszowej (nie tworząc nowego akapitu), lub wpisz `list-item`, aby wyświetlić element w taki sposób, jak gdyby był on umieszczony wewnątrz elementu `li` (zobacz strony od 203 – 212).

Wskazówki

- Netscape 6 jest tak skrupulatny, gdy chodzi o przestrzeganie standardów, że dodaje nieco wolnej przestrzeni poniżej wewnątrzwierszowych obrazków umieszczanych w komórkach tabel. Trzeba o tym pamiętać. Jeśli w komórce tabeli znajduje się tylko jeden obrazek, można uniknąć tego efektu, zmieniając obrazek w element blokowy.
- W razie użycia właściwości `display: none` element zostanie ukryty, nie pozostawiając żadnego śladu. Miejsce po nich zostaje zajęte przez inne elementy strony.
- Właściwość `display` nie jest dziedziczona.
- Do ukrywania elementów bez jednoczesnego usuwania ich z układu strony można użyć właściwości `visibility: hidden`. W takim przypadku element staje się niewidoczny, lecz zajmowany przez niego obszar wciąż jest uwzględniany w układzie strony.

Wyświetlanie i ukrywanie elementów

Rysunek 11.6. Wykorzystamy właściwość `display`, aby zmienić na elementy blokowe elementy a umieszczone w dziale nawigacyjnym, dzięki czemu każdy z nich zostanie wyświetlony w osobnym wierszu. Zwróć uwagę, iż zmieniam właściwość `display` wyłącznie w elementach a umieszczonych w dziale `navigation`; te same elementy umieszczone w innych częściach strony nie zostaną w żaden sposób zmodyfikowane



Rysunek 11.7. Teraz elementy a są wyświetlane w osobnych wierszach. Zwróć uwagę, że element a umieszczony w pierwszym wierszu ostatniego widocznego akapitu („budowanie stajni”) wciąż jest wyświetlany jako element wewnątrzwierszowy (czyli tak, jak powinien być wyświetlany)

```

body {position: absolute; top: 250px; left: 2%;}
#content {position: absolute; top: 8px; left: 30%;}
#navigation {position: absolute; top: 10px; left: 2%;}
#navigation a {display: block;

```

Rysunek 11.8. Choć podając przesunięcia należy dokładnie określić, gdzie element umiejscawiany bezwzględnie ma zostać wyświetlony (oczywiście w odniesieniu do jego elementu nadrzędnego), można jednak do tego celu wykorzystać wartości procentowe, tworząc w ten sposób elastyczny układ, który może się dostosowywać do wielkości okna przeglądarki



Rysunek 11.9. Strona szybko nabiera kształtu. Dział zawartości (content) zaczyna się w odległości 30% szerokości okna przeglądarki od jego lewej krawędzi, a obszar nawigacyjny (navigation) rozpoczyna się 10 pikseli poniżej górnej krawędzi okna przeglądarki i tylko 2% od jego lewej krawędzi. Obszar bg został chwilowo wyświetlony poniżej obszaru nawigacyjnego

Bezwzględne rozmieszczanie elementów

Wszystkie elementy strony są rozmieszczane na niej w takim porządku, w jakim się pojawiają. Oznacza to, że jeśli na przykład znacznik `img` znajduje się przed znacznikiem `p`, to obrazek zostanie wyświetlony przed akapitem. Nazywane jest to naturalnym układem strony. Możesz jednak zmienić naturalny układ strony, rozmieszczając elementy **bezwzględnie** — czyli określając ich położenie względem elementu nadrzędnego.

Aby rozmieścić element bezwzględnie:

1. Wpisz `position: absolute;` (pamiętaj o średniku, odstęp po nim jest już opcjonalny).
2. Wpisz `top`, `right`, `bottom` lub `left`.
3. Podaj `tv`, gdzie `v` określa przesunięcie elementu w stosunku do jego elementu nadrzędnego, wyrażone w formie wartości bezwzględnej lub względnej (na przykład `10px` lub `2em`), albo w formie wartości procentowej, określonej względem elementu nadrzędnego.
4. Jeśli chcesz, powtórz czynności z punktów 2. i 3. dla innych kierunków, rozdzielając poszczególne pary właściwość-wartość średnikami.

Wskazówki

- Pamiętaj, że elementy są rozmieszczane względem ich elementu nadrzędnego. Informacje o elementach nadrzędnych można znaleźć na stronie 28.
- W przypadku tworzenia układów elastycznych, dostosowujących się do wielkości okna przeglądarki, należy posługiwać się wartościami procentowymi.
- Ponieważ elementy rozmieszczane bezwzględnie nie są uwzględniane przy naturalnym rozmieszczaniu elementów strony, mogą zatem wzajemnie na siebie zachodzić (co nie zawsze jest niepożądane).
- Jeśli nie określisz przesunięcia elementu, zostanie on wyświetlony w swoim naturalnym położeniu, ale nie będzie miał wpływu na położenie kolejnych elementów.
- Sposób pozycjonowania elementu nie jest dziedziczony w jego elementach podrzędnych.

Określanie stałego położenia elementu w oknie przeglądarki

Gdy użytkownik przewija zawartość okna przeglądarki, porusza się ona w górę i w dół, choć na przykład przyciski *Wstecz* i *Do przodu* pozostają nieaktywne. Kaskadowe arkusze stylów pozwalają na określenie stałego położenia elementów względem okna przeglądarki, dzięki czemu elementy te nie są przesuwane w przypadku przewijania zawartości strony.

Aby określić stałe położenie elementu w oknie przeglądarki:

1. Wpisz `position: fixed;` (nie zapomnij średnika).
2. Wpisz `top`, `right`, `bottom` lub `left`.
3. Wpisz `lv`, gdzie `v` określa przesunięcie elementu względem krawędzi okna przeglądarki, wyrażone w formie wartości bezwzględnej lub względnej (na przykład `10px` lub `2em`), albo w formie wartości procentowej, określanej względem wielkości okna przeglądarki.
4. W razie potrzeby powtórz kroki 2. oraz 3., definiując przesunięcia w innych kierunkach. Pamiętaj, aby poszczególne pary właściwość-wartość oddzielać średnikami.

Wskazówki

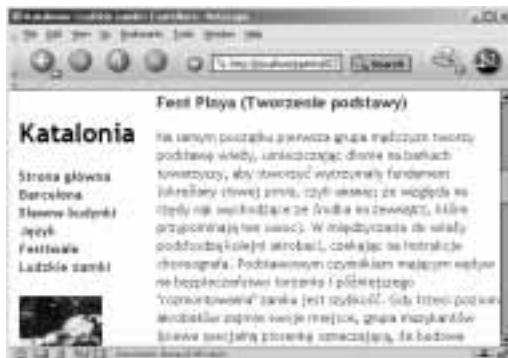
- Pamiętaj, że przesunięcia elementów o ustalonym położeniu są określane względem okna przeglądarki, natomiast przesunięcia elementów umiejscawianych bezwzględnie — w odniesieniu do ich elementu nadrzędnego.
- Gdyby umiejscawianie ustalone było lepiej obsługiwane w przeglądarkach, stanowiłoby ono doskonały substytut ramek i układów ramek.
- Niestety przeglądarka Internet Explorer przeznaczona do użycia w systemach Windows (włącznie z wersją 6) nie obsługuje tej metody umiejscawiania elementów.
- Sposób umiejscowienia elementu nie jest dziedziczony.

```
#bg {position: fixed; top: 250px; left: 24px;}
#content {position: absolute; top: 0px; left: 304px;}
#navigation {position: fixed; top: 10px; left: 24px;}
#navigation {display: block;}
```

Rysunek 11.10. Jedyną różnicą pomiędzy przedstawionymi tu regułami stylów a regułami z rysunku 11.8, zamieszczonego na stronie 179, jest użycie właściwości `position` z wartością `fixed` w regułach dla stylów `bg` oraz `navigation`



Rysunek 11.11. Na pierwszy rzut oka ta strona wygląda tak samo, jak strona z rysunku 11.9, przedstawionej na stronie 179



Rysunek 11.12. Jednak gdy użytkownik zacznie przewijać stronę, okaże się, że zarówno pasek nawigacyjny, jak i jego tło nie zmieniają położenia

```
#h2 {position: absolute; top: 50px; left: 25px;}
#content {position: absolute; top: 0px; left: 300px;}
#navigation {position: absolute; top: 50px; left: 25px;}
#navigation & {display: block;}

h2, h3 {position: relative; left: -25px;}
```

Rysunek 11.13. Ten fragment kodu przesuwa nagłówki h2 oraz h3 o 25 pikseli w lewo względem miejsca, w którym powinny się znajdować



Rysunek 11.14. Nagłówki h2 oraz h3 są przesunięte w lewo poza obszar działu strony, w jakim są umieszczone. Nie ma to jednak żadnego znaczenia — są one przesuwane względem swego oryginalnego położenia, niezależnie od tego, gdzie by się nie znajdowały

Przesuwanie elementów względem ich naturalnego położenia

Każdy element posiada pewne naturalne położenie w zawartości dokumentu. Przesuwanie elementów względem tego położenia nazywane jest *określanie położenia względnego*. W przypadku wykorzystania tego sposobu umieszczania położenie elementów sąsiednich nie jest w żaden sposób modyfikowane.

Aby przesunąć element względem jego naturalnego położenia:

1. Wpisz `position: relative;` (nie zapomnij o średniku).
2. Wpisz `top`, `bottom`, `left` lub `right`.
3. Wpisz `:v`, gdzie `v` określa przesunięcie elementu względem jego naturalnego położenia, wyrażone w formie wartości bezwzględnej lub względnej (na przykład `10px` lub `2em`).
4. W razie potrzeby powtórz kroki 2. oraz 3., definiując przesunięcia w innych kierunkach. Pamiętaj, aby poszczególne pary właściwość-wartość oddzielać średnikami.

Wskazówki

- W przypadku tego sposobu umieszczania słowo „względem” odnosi się do oryginalnego położenia elementu, a nie elementów sąsiadujących z nim. Nie można przesunąć elementu względem położenia innego elementu strony. Zamiast tego element można przesunąć względem położenia, jakie powinien on zajmować. Tak, to jest bardzo ważne!
- Położenie innych elementów nie jest w żaden sposób modyfikowane — są one rozmieszczane w naturalny sposób, tak jak gdyby przesunięty element znajdował się w swym *oryginalnym* położeniu. Dlatego też elementy mogą się wzajemnie przesłaniać.
- Podane przesunięcie nie będzie miało żadnego wpływu na położenie elementu, jeśli w stylu nie zostanie określona właściwość `position`.
- Sposób umieszczenia elementu nie jest dziedziczony.

Modyfikowanie tła

Temin „tło” odnosi się nie do tła całej strony, lecz do tła konkretnego elementu. Innymi słowy, można określać tło dowolnego elementu — w tym także obrazków, pól formularzy oraz tabel.

Aby zmienić kolor tła:

1. Wpisz `background-color:`.
2. Wpisz `transparent` (aby widoczne było tło elementu nadrzędnego) lub `color`, gdzie `color` jest nazwą koloru lub wartością rgb (patrz strona 46 lub tylna okładka książki).

Aby zastosować obraz tła:

1. Wpisz `background-image:`.
2. Wpisz `url (obrazek.gif)`, gdzie `obrazek.gif` jest nazwą obrazka, który powinien być wyświetlony jako tło elementu. Aby żaden obraz nie był wyświetlany, wpisz `none` (rysunki 11.17 oraz 11.18).

Aby powtórzyć obraz tła:

1. Wpisz `background-repeat:` kierunek, gdzie kierunek może przybierać następujące wartości: `repeat` (aby obrazek był powtarzany zarówno w pionie, jak i w poziomie), `repeat-x` (aby obrazek był powtarzany w poziomie), `repeat-y` (aby obrazek był powtarzany w pionie) lub `no-repeat` (aby obrazek nie był powtarzany).

Aby określić, czy obrazek tła ma ustalone położenie:

1. Wpisz `background-attachment:`.
2. Wpisz `fixed`, aby położenie obrazka tła w oknie przeglądarki było stałe, lub `scroll`, aby obrazek przesuwał się wraz z przewijaną zawartością okna przeglądarki.

Aby określić położenie obrazu tła elementu:

1. Wpisz `background-position:` `x y`, gdzie `x` oraz `y` mogą być wyrażone w formie wartości procentowych lub bezwzględnych. Jako `x` można także użyć wartości predefiniowanych `left`, `center` lub `right`, a jako `y` wartości `top`, `center` lub `bottom`.

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

h1 {background-color: #fff;}

bg {position: absolute; top: 250px; left: 2%;
background-color: black;}

#content {position: absolute; top: 0px; left: 10%;
background-color: #fff;}

#navigation #home {position: absolute; top: 10px;
left: 2%; background-color: #fff;}

#navigation #display: block;

#navigation #home: hover {background-
color: transparent;}
```

Rysunek 11.15. Definiujemy białe tło elementu `body`, działu treści oraz wskazanych myszką łączą nawigacyjnych. Dział `bg` strony ma mieć czarne tło (to przygotowanie do zmian wprowadzanych na stronach 190–192). Wskazane myszką łącze do bieżącej strony ma być przezroczyste, gdyż nie chcemy, aby wyglądało ono jak typowe łącze



Rysunek 11.16. Prawdopodobnie najtrudniejszą do zauważenia modyfikacją jest określenie białego tła strony. Niemniej jednak jest to modyfikacja kluczowa (o czym przekonamy się na stronach od 190 – do 192)

```
body {background-image: url(img/castle_bg.jpg); repeat-x
background-color: #fff;}
h1 {background-color: #fff;}
img#img {position: absolute; top: 100px; left: 10px;
background-color: black;}
#content {position: absolute; top: 0px; left: 80px;
background-color: #fff;}
#navigation ul li {position: absolute; top: 10px;
left: 10px; background-color: #fff;}
#navigation ul {display: block;}
#navigation ul li a {background-color: transparent}
```

Rysunek 11.17. Tylko na chwilę wykorzystamy obraz tła, którego w praktyce trudno jest używać efektywnie



Rysunek 11.18. Pomimo tego, że obraz tła elementu body jest powtarzany w poziomie, jest on przesłaniany przez biały obraz tła działu content (który, dzięki zastosowaniu atrybutu id, jest określany z większą precyzją). Zauważ, że obraz tła rozpoczyna się w lewym, dolnym rogu całej strony, a nie ekranu (jak jest to błędnie interpretowane w IE dla komputerów Macintosh)

Aby określić wszystkie właściwości tła w jednym miejscu:

1. Wpisz background:.
2. Wpisz dowolną kombinację akceptowanych wartości związanych z określaniem postaci tła elementu (opisanych na poprzedniej stronie). Wartości te mogą być zapisane w dowolnej kolejności.

Wskazówki

- Domyślną wartością właściwości background-color jest transparent. Domyślną wartością właściwości background-image jest none. Domyślną wartością właściwości background-repeat jest repeat, właściwości background-attachment — scroll, a właściwości background-position — top left.
- Używając skróconej właściwości background (opisanej na początku tej strony), nie trzeba podawać wszystkich wartości określających postać tła elementu. Należy jednak pamiętać, że wszystkie właściwości, które nie zostaną jawnie określone, przyjmą swoje wartości domyślne (co może przesłonić reguły stylów zdefiniowane wcześniej).
- Właściwości background nie są dziedziczone. Chcąc przesłonić inne reguły stylów, należy jedynie jawnie podać wartości domyślne, takie jak transparent lub scroll.
- W przypadku wykorzystania powtarzanego (repeat) obrazu tła i jednoczesnego użycia właściwości background-position będzie ona określać położenie pierwszego z powtarzanych obrazów tła.
- Aby stworzyć tło całej strony, należy określić tło elementu body (to jedyna metoda, aby wykorzystać właściwość background w Netscape 4).
- W przypadku określenia zarówno koloru, jak i obrazu tła, kolor zostanie wykorzystany do momentu pobrania obrazu tła, a potem będzie widoczny przez jego przezroczyste obszary (jeśli kolor nie zostanie określony, to Netscape 4 użyje koloru czarnego).

Zmiana koloru

Można zmieniać kolor dowolnego elementu, w tym także tekstu, linii poziomych oraz pól formularzy.

Aby zmienić kolor:

1. Wpisz `color`.
2. Wpisz nazwa_koloru, gdzie nazwa_koloru to jedna z 16 predefiniowanych nazw kolorów (patrz strona 46 lub tylna okładka książki),
lub wpisz `#rrggbb`, gdzie `rrggbb` to szesnastkowa reprezentacja koloru (patrz strona 46 lub tylna okładka książki),
lub wpisz `rgb(r, g, b)`, gdzie `r, g i b` to liczby całkowite z zakresu od 0 do 255, określające odpowiednio ilość koloru czerwonego, zielonego oraz niebieskiego w kolorze wynikowym, który chcemy zastosować,
lub wpisz `rgb(r%, g%, b%)`, gdzie `r, g i b` to wartości określające procentową ilość czerwonego, zielonego i niebieskiego w kolorze wynikowym.

Wskazówki

- Jeśli jako `r, g` lub `b` wpiszesz wartość większą od 255, zostanie ona zastąpiona liczbą 255. Podobnie, jeśli jako wartość procentową wpiszesz więcej niż 100%, zostanie ona zastąpiona wartością 100%.
- Właściwość `color` najczęściej jest stosowana do określania koloru tekstu. Więcej informacji na ten temat można znaleźć na stronie 166, w sekcji „Definiowanie koloru tekstu”.
- Zmienianie koloru obrazu nie ma żadnego widocznego efektu (taką operację należałoby wykonać w programie graficznym). Można jednak zmienić kolor tła elementu `img` (który będzie widoczny przez przezroczyste obszary obrazka). Więcej informacji na ten temat można znaleźć na stronie 182, w sekcji „Zmiana koloru tła tekstu”.

```
body {background-color:#fff; color:#000;
#bg {position: absolute; top: 150px; left: 1%;
background-color: #000}

#content {position: absolute; top: 0px; left: 30%;
background-color: #fff;

#navigation {position: absolute; top: 10px; left: 5%;
color: white;
#navigation a {display: block; color: white;

#navigation a:link, #navigation
a:visited, #navigation a:current, {color: #fff;}
#navigation a:hover {background-color: #fff; color: #339;

#navigation a:hover:active {
background-color: transparent; color: #fff;

h2 {background-color: #339; color: #fff;
```

Rysunek 11.19. Zmieniłam domyślny kolor całego tekstu na stronie na czarny (w elemencie `body`). Jednak tekst w obszarze nawigacyjnym będzie biały (aby odróżniał się od przyszłego tła tego obszaru, które zostanie zmienione na stronach 190 – 192). Łącze do bieżącej strony umieszczone w obszarze nawigacyjnym będzie białozłote (#339, tak aby nie wyglądało jak łącze). Jednak inne łącza, po wskazaniu ich wskaźnikiem myszy, powinny być wyświetlane na niebiesko (aby były wyraźnie widoczne na białym tle strony, zdefiniowanym na stronie 182)



Rysunek 11.20. Teraz, gdy łącza nawigacyjne są białe, można je zobaczyć wyłącznie w przypadku umieszczenia na nich wskaźnika myszy (lub zdefiniowania na stronie jakiegoś tła — patrz strony 190 – 192)

```
body {background-color:#fff; color:#000;}
#bg {position: absolute; top: 20px; left: 20px;
background-color:#fff;}

#content {position: absolute; top: 0px; left: 0px;
background-color:#fff;}

#navigation {position: absolute; top: 0px; left: 20px;
color:white;}
#navigation a {display: block; color:white;}

#navigation a:link, #navigation
a:visited, #navigation
a:current, #navigation
a:active {background-color:#fff; color:#fff;}

#navigation a:current {
background-color:#fff;
color:#fff; cursor:default;}

h1 {background-color: #000; color: #fff;}
```

Rysunek 11.21. Dodałam właściwość `cursor` do selektora wybierającego wyłącznie łącza klasy `current`, należące do elementu o identyfikatorze `navigation`. Co więcej, łącza te muszą być aktualnie wskazywane myszką. Całkiem interesujący selektor, nieprawdaż?



Rysunek 11.22. Teraz, gdy ktoś wskaże łącze do bieżącej strony, nie będzie ono wcale wyglądać jak łącze (choć adres strony docelowej będzie wyświetlany w pasku stanu, a samo łącze będzie działać poprawnie)

Modyfikacja wskaźnika myszy

Zazwyczaj to przeglądarka określa wygląd wskaźnika myszy, przy czym w większości przypadków ma on kształt strzałki, a po umieszczeniu go w obszarze łącza — kształt dłoni z wyprostowanym palcem wskazującym. Dzięki CSS to projektant strony może określić postać wskaźnika myszy.

Aby zmienić kształt wskaźnika myszy:

1. Wpisz `cursor:`.
2. Wpisz `pointer`, aby wskaźnik miał kształt dłoni (☞), `default`, aby miał kształt strzałki (↖), `crosshair` (+), `move` (↔), `wait` (⌚), `help` (??) lub `text` (I),
lub wpisz `auto`, aby postać wskaźnika była określana automatycznie, w zależności od sytuacji,
lub wpisz `x-resize`, aby wskaźnik przybrał postać dwukierunkowej strzałki, przy czym `x` jest określeniem jednego z głównych kierunków geograficznych, na przykład `n` (*north* — północ), `nw` (*northwest* — północny zachód), `e` (*east* — wschód).
Przykładowo, wskaźnik `e-resize` ma postać ↔.

Wskazówki

- Internet Explorer 5.x dla Windows rozpoznaje wartość `hand`, lecz nie rozpoznaje standardowej wartości `pointer` (☞). Aby zadowolić tych, którzy lubią wszystko robić po swojemu, Microsoft Internet Explorer 6 oraz inne przeglądarki działające zgodnie ze standardami (takie jak Netscape 6) obsługują **obie** wartości — `cursor:pointer` i `cursor:hand`. Kolejność zapisu jest w tym przypadku ważna, gdyż przeglądarka IE5 dla Windows wybierze wartość `pointer`, jeśli ta zostanie zapisana jako ostatnia.
- Rysunki przedstawione na tej stronie zostały sporządzone przy użyciu przeglądarki IE6 w systemie Windows 98. Wskaźniki używane w różnych wersjach przeglądarek oraz w różnych systemach operacyjnych nieznacznie się od siebie różnią. Przykładowo, wskaźnik typu `wait` na komputerach Macintosh ma kształt zegarka:
- Osobiście uważam, że nazwy używane do określania postaci wskaźnika myszy są dosyć mylące. Wartość `default` (ang. domyślny) wcale nie określa domyślnego kształtu wskaźnika myszy, lecz przypomina strzałkę, dlatego nazwałabym ją `pointer` (ang.: wskaźnik); jednak wartości `pointer` odpowiada wskaźnik o kształcie dłoni (ang.: *hand*), z kolei wartość `hand` jest niestandardowa i stosowana tylko w Internet Explorerze. O rany!

Tworzenie obramowań

Można stworzyć obramowanie wokół elementu, a następnie określić jego grubość, styl oraz kolor. W przypadku określenia wypchnięcia (patrz strona 189) obramowanie będzie otaczać zarówno zawartość danego elementu, jak i jego wypchnięcie.

Aby zdefiniować styl obramowania:

1. Wpisz `border-style: typ`, gdzie `typ` może przyjmować jedną z następujących wartości: `none`, `dotted`, `dashed`, `solid`, `double`, `groove`, `ridge`, `inset` lub `outset`.

Aby określić grubość obramowania:

1. Wpisz `border-width: n`, gdzie `n` określa grubość obramowania i jest liczbą wraz z jednostkami (na przykład `4px`).

Aby określić kolor obramowania:

1. Wpisz `border-color: kolor`, gdzie `kolor` jest nazwą koloru lub wartością rgb (patrz strona 46 lub tylna okładka książki).

Aby jednocześnie określić jedną lub kilka właściwości obramowania:

1. Wpisz `border`.
2. Jeśli chcesz, wpisz `-top`, `-right`, `-bottom` lub `-left`, aby ograniczyć efekt działania reguły do konkretnej krawędzi obramowania.
3. Jeśli to konieczne, wpisz właściwość, gdzie właściwość to `style`, `width` lub `color`. W ten sposób możesz ograniczyć działanie reguły do konkretnej właściwości.
4. Wpisz `:`.
5. Wpisz odpowiednie wartości (opisane na początku strony). Jeśli pominąłeś krok 3., możesz podać dowolną lub wszystkie trzy właściwości krawędzi (na przykład `border: 1px solid` lub `border-right: 2px dashed green`). Jeśli w kroku 3. określiłeś wybraną właściwość, powinieneś podać wyłącznie jedną z wartości odpowiednich dla tej właściwości (na przykład `border-right-style: dotted`).

```
body {background-color:#fff; color:#000}
#by {position: absolute; top: 250px; left: 20px;
background-color:black}

#comment {position: absolute; top: 0px; left: 30px;
background-color:#fff;border-left: 1px solid black;
}

#navigation {position: absolute; top: 10px; left: 20px;
color:white}
#navigation a {display: block; color:#fff}

#navigation a:link, #navigation
a:visited, #navigation a:current, {color:#fff;}
#navigation a:hover {background-color:#fff; color:#333}

#navigation a:lower, #navigation
a:current {background-color:transparent; color:#fff;
cursor:default}

h1 {background-color: #333; color: #fff}
```

Rysunek 11.23. Przy wykorzystaniu skrótovej właściwości `border-left` można jednocześnie określić wszystkie trzy właściwości lewej krawędzi obramowania elementu



Rysunek 11.24. Oto proste obramowanie o szerokości 1 piksela i postaci linii ciągłej. Zwróć uwagę, że pomiędzy obramowaniem i tekstem elementu przydałoby się wyświetlić jakiś odstęp. Już wkrótce się tym zajmiemy (patrz strona 188)

```
div {height: 70px; width: 200px; margin-left: 10px;
border-color: black; border-width: 2px 3px 10px 3px;
background: #fff;
border-style: dotted;
border-style: dashed none;
border-style: none double;
border-top-style: groove;
border-right-style: ridge;
border-bottom-style: inset;
border-left-style: outset}
```

Rysunek 11.25. Za pomocą jednej właściwości `border-color` określiłam kolory wszystkich czterech krawędzi obramowania. Następnie, korzystając z właściwości `border-width`, określiłam grubości poszczególnych krawędzi (zaczynając od wartości 2px i przesuając się zgodnie z ruchem wskazówek zegara). W końcu stworzyłam klasy dla każdego z istniejących stylów i przypisałam je jednej lub kilku krawędziom poszczególnych działów strony



Rysunek 11.26. Zwróć uwagę, jak działają różne właściwości skrótowe. Na przykład `border-style: dotted` (pierwszy przykład) tworzy kropkowane obramowanie na wszystkich czterech krawędziach, natomiast `border-style: dashed none` tworzy obramowanie w postaci linii przerywanej na górnej i dolnej krawędzi elementu i nie wyświetla żadnego obramowania na krawędzi prawej i lewej. Przeglądarki (IE 6 z lewej oraz Netscape 7 z prawej) w nieco odmienny sposób interpretują poszczególne style obramowań

Wskazówki

- Postać obramowania nie jest dziedziczona.
- Poszczególne właściwości obramowania (`border-width`, `border-style` oraz `border-color`) mogą przyjmować od jednej do czterech wartości. W razie podania jednej wartości określi ona postać wszystkich krawędzi obramowania. Jeśli podany dwie wartości, pierwsza z nich określać będzie postać dolnej i górnej krawędzi, a druga — krawędzi prawej i lewej. W razie podania trzech wartości pierwsza z nich posłuży do określenia górnej krawędzi, druga — krawędzi prawej i lewej, a trzecia — dolnej. W razie podania czterech wartości zostaną one użyte do określenia postaci odpowiednio górnej, prawej, dolnej i lewej krawędzi obramowania.
- Aby obramowanie było wyświetlone, należy określić przynajmniej jego styl. Jeśli styl nie zostanie określony, obramowanie nie będzie widoczne. Domyślną wartością stylu jest `none`.
- Jeśli skorzystamy z właściwości skrótowych (takich jak `border` lub `border-left`), a wartości nie zostaną podane jawnie, przyjmą one wówczas odpowiednie wartości domyślne. A zatem właściwość `border: 1px black` odpowiada właściwości `border: 1px black none` (nawet jeśli styl został wcześniej określony za pomocą właściwości `border-style`).
- Domyślny kolor obramowania odpowiada wartości właściwości `color` danego elementu (patrz strona 166).
- Grubość obramowania można także określić za pomocą wartości ogólnych — `thin` (cienka), `medium` (średnia) oraz `thick` (gruba). Wartością domyślną jest `medium`, która w Internet Explorerze wynosi 4 piksele, a w Netscape 3 piksele.
- Starsze wersje przeglądarek (wcześniejsze niż IE 5.5 oraz Netscape 6) nie obsługują wszystkich istniejących, pojedynczych właściwości, określających postać obramowania.
- Właściwość `border` można stosować w tabelach oraz w komórkach table. Więcej informacji na ten temat można znaleźć na stronie 218, w sekcji „Dodawanie krawędzi tabel”.
- Szczerze mówiąc, uważam, że ktoś przesadził z ilością możliwych sposobów określania postaci obramowań.

Dodawanie wypełnienia wokół elementu

Wypełnienie to dodatkowa przestrzeń wokół elementu, umieszczona wewnątrz jego obramowania. Można je sobie wyobrazić jako okazałych rozmiarów brzuch Świętego Mikołaja, opięty jego pasem. W przypadku wypełnienia można określać wyłącznie jego szerokość, nie ma natomiast możliwości określania ani jego koloru, ani faktury.

Aby dodać wypełnienie wokół elementu:

1. Wpisz `padding: x`, gdzie `x` jest szerokością dodawanego wypełnienia, wyrażoną za pomocą konkretnych jednostek (10px) lub w formie wartości procentowej, odnoszącej się do wielkości elementu nadrzędnego (20%).

Wskazówki

- Jeśli w definicji wypełnienia zostanie podana tylko jedna wartość, to posłuży ona do określenia szerokości wypełnienia ze wszystkich czterech stron elementu. W przypadku podania dwóch wartości pierwsza z nich posłuży do określenia wartości wypełnienia u góry i u dołu elementu, a druga — z jego lewej i prawej strony. W razie podania trzech wartości pierwsza z nich określać będzie wypełnienie u góry elementu, druga — z lewej i prawej strony elementu, a trzecia — u dołu elementu. W przypadku podania czterech wartości posłużą one do określenia odpowiednio górnego, prawego, dolnego i lewego wypełnienia elementu.
- Do właściwości `padding` można także dodać jedną z następujących końcówek, określających postać wypełnienia z konkretnej strony elementu: `-top`, `-bottom`, `-left`, `-right`. Pomiedzy słowem `padding` a końcówką nie powinno być żadnych dodatkowych odstępów.
- Wypełnienie nie jest dziedziczone.
- W przypadku tworzenia elastycznych układów, które rozszerzają się lub zmniejszają w zależności od wielkości okna przeglądarki, wypełnienie można określać, wykorzystując wartości procentowe lub jednostki em.

```
body {background-color:#fff; color:#000;
padding:0 0 0 0}
#bg {position: absolute; top: 250px; left: 28;
background-color: black}

#content {position: absolute; top: 10px; left: 10px;
background-color: #fff; border-left: 1px solid black;
padding: 20px;}

#navigation {position: absolute; top: 10px; left: 24;
color: white}
#navigation a {display: block; color: #fff;
padding-left: 10px; padding-right: 10px;}

[...]
```

```
h1 {background-color: #333; color: #fff;
padding: 0 10px;}
```

Rysunek 11.27. Przypisanie wartości 0 właściwości `padding` nie jest złym pomysłem, gdyż w różnych przeglądarkach domyślne wypełnienia różnych elementów mają różne wielkości. Zwróć uwagę, że ostatnie dwa style wypełnienia (dla elementów `#navigation` oraz `h1`) są równoważne



Rysunek 11.28. Zawartość elementu wygląda znacznie lepiej, gdy pomiędzy nią a obramowaniem elementu znajduje się jakaś wolna przestrzeń. Zwróć uwagę, że w przypadku dodania wypełnienia tło obramowania („Katalonia”) jest powiększane

```
body {background-color:#fff; color:#000;
padding:0 0 0 0;margin:0 0 0 0;
#bg {position: absolute; top: 250px; left: 25px;
background-color:#000;}

#content {position: absolute; top: 0px; left: 20px;
background-color:#fff;border-left: 1px solid black;
padding: 20px; }

[...]
```

```
h3 {background-color: #000; color: #fff;
padding: 0 10px;margin-top: 10px;}
h3 {margin: 0;}
h3 {margin: 10px 0 0 0;}
h3 {margin: 10px 0 0 0;}
```

Rysunek 11.29. Margines elementu `h3` moglibyśmy także określić za pomocą właściwości `margin-top: 15px`



Rysunek 11.30. Tekst prezentowany w dziale `content` strony wygląda znacznie lepiej, gdy pomiędzy nagłówkiem a akapitem jest mniej wolnego miejsca. Pamiętaj, że margines to obszar pomiędzy prostokątami elementów, których rozmiar jest z kolei określany przez tak różnorodne czynniki, jak zawartość, wysokość wiersza, właściwość `height` i tak dalej. Zwróć także uwagę, że margines pomiędzy elementem `h3` oraz `p` ma wielkość 15 pikseli (czyli odpowiada większemu z marginesów, a nie ich sumie, która ma wartość 20 pikseli)

Określanie marginesów wokół elementu

Margines to przezroczysta przestrzeń, oddzielająca elementy od siebie. Marginesy są niezależne oraz znajdują się na zewnątrz wypełnienia (patrz strona 188) i obramowania (patrz strona 186) — można by je porównać do przestrzeni osobistej Świętego Mikołaja.

Aby określić marginesy elementu:

1. Wpisz `margin: x`, gdzie `x` jest szerokością dodawanego obszaru, wyrażoną jako długość lub wartość procentowa określana względem szerokości elementu nadrzędnego, bądź też przyjmującą wartość `auto`.

Wskazówki

- Właściwości `margin` można przypisać od jednej do czterech wartości — patrz pierwsza rada dotycząca właściwości `padding` (na stronie 188).
- Do właściwości `margin` można także dodać jedną z czterech końcówek: `-top`, `-bottom`, `-left` lub `-right`, aby ograniczyć margines do konkretnej strony elementu. Pomiędzy słowem `margin` a końcówką nie powinno być żadnego odstępu (na przykład `margin-top: 10px`).
- Faktyczna wielkość wartość `auto` przypisywanej właściwości `border` zależy od wartości właściwości `width` (patrz strona 190). Jeśli przypiszemy wartość `auto` jednemu marginesowi elementu, przeglądarka nada mu największą z możliwych wartości. Jeśli jednocześnie przypiszemy lewemu i prawemu marginesowi wartości `auto`, to przeglądarka nada im dwie największe, równe wartości. Fakt ten można wykorzystać w celu wyśrodkowania elementu na stronie.
- Jeśli jeden element jest umieszczony powyżej drugiego, wykorzystany zostanie tylko jeden z dwóch stykających się ze sobą marginesów — ten większy. W takim przypadku mówimy, że mniejszy margines *zapada się*. Ten sposób działania nie dotyczy marginesów prawych i lewych.
- Marginesy nie są dziedziczone.

Określanie wysokości i szerokości elementu

Istnieje możliwość określenia wysokości i szerokości niemal wszystkich elementów strony, w tym także obrazków, pól formularzy, a nawet bloków tekstu.

Aby określić wysokość i szerokość elementu:

1. Wpisz `width: w`, gdzie `w` określa szerokość obszaru elementu, wyrażoną w formie długości (liczby wraz z określeniem jednostki) lub wartości procentowej (obliczanej względem szerokości elementu nadrzędnego). Możesz także użyć wartości `auto`, aby szerokość elementu została określona przez przeglądarkę.
2. Wpisz `height: h`, gdzie `h` określa wysokość obszaru elementu i może być wyrażona wyłącznie w formie długości (liczby wraz z określeniem jednostki). Możesz także użyć wartości `auto`, aby wysokość elementu została określona przez przeglądarkę.

Wskazówki

- Jeśli wysokość i szerokość elementu nie zostanie jawnie podana, przeglądarka użyje wartości `auto` (patrz strona 191).
- Pamiętaj, że wartość procentowa jest określana na podstawie szerokości elementu nadrzędnego — a nie oryginalnej szerokości samego elementu.
- Wypełnienie, obramowania oraz marginesy nie są uwzględniane w wartości właściwości `width` (z wyjątkiem przeglądarki IE 5.x dla Windows, która błędnie zakłada, że właściwość `width` jest sumą szerokości obszaru zawartości elementu, jego wypełnienia i obramowania — patrz następna strona).
- Szerokość i wysokość elementu nie są dziedziczone.
- Istnieją także właściwości `min-width`, `min-height` oraz `max-width` i `max-height`, lecz aktualnie nie są one obsługiwane przez wszystkie przeglądarki.

```
<div id="bg">
  
</div>
```

Rysunek 11.31. Przypomnij sobie, że w kodzie (X)HTML dokumentu określiliśmy szerokość obrazka umieszczonego w dziale `bg` strony — 100 pikseli

```
body {background-color:#fff; color:#000;
padding:0 0 0 0;margin:0 0 0 0}
#bg {position: absolute; top: 150px; left: 15px;
background-color:#f0f0f0; top:0; left:10px;
margin-left:-50px;}

#bg img {width:550px; height:650px;}

#content {position: absolute; top: 0px; left: 30px;
background-color:#fff;border-left: 1px solid black;
padding: 10px;}

#navigation {position: absolute; top: 10px; left: 15px;
color: white; width: 200px;
background-color: black; height: 30px;
padding-left: 10px; padding-right: 10px;}
```

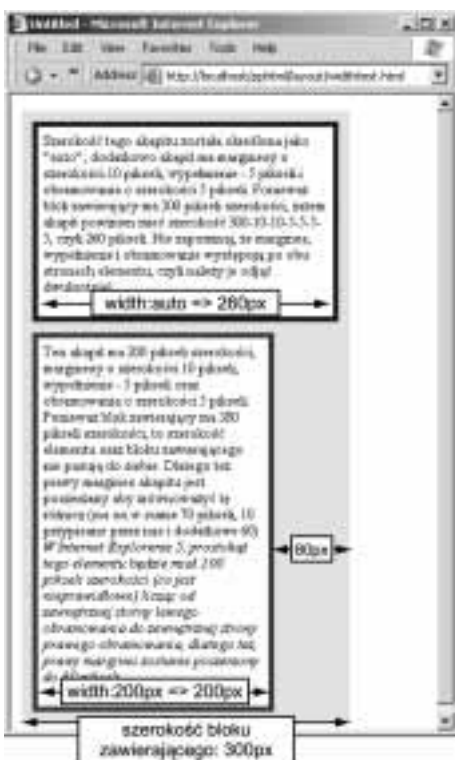
Rysunek 11.32. Określiłam szerokość obrazka umieszczonego w dziale `bg` przy użyciu właściwości `width` (przesłaniając w ten sposób wartość atrybutu (X)HTML, przedstawionego na rysunku 31). Zwróć uwagę, że zmieniłam także pionowe i poziome przesunięcie elementu `div` o identyfikatorze `bg`, a dodatkowo przypisałam ujemną wartość lewemu marginesowi elementu, aby wyrównać tę część obrazka, która mnie interesuje



Rysunek 11.33. Zwróć uwagę, że dział zawartości zaczynający się od 30% szerokości strony, licząc od jej lewej krawędzi, zachodzi na dział `bg` strony. Ponieważ dział zawartości strony ma białe tło, jego część pokrywająca się z tym obszarem będzie niewidoczna. Dodatkowo zwróć uwagę, iż także część nawigacyjna strony jest w tej chwili niewidoczna. Już wkrótce rozwiążemy ten problem

```
div {width:300px; height:200px;background: yellow;
padding: 5px; border: 5px solid blue;
p {width:200px; background: white; margin:10px;
padding: 5px; border: 5px solid purple;
```

Rysunek 11.34. Ustawiłam, że nadrzędny element `div` będzie miał szerokość 300 pikseli. To będzie nasz obszar zawierający. Oba umieszczone w nim akapity mają marginesy o szerokości 10 pikseli, wypełnienie o szerokości 5 pikseli i obramowanie o szerokości 5 pikseli z każdej strony. Szerokość pierwszego akapitu jest określana automatycznie, natomiast drugiemu jawnie nadałam szerokość 200 pikseli



Rysunek 11.35. Jeśli szerokość jest określana automatycznie (jak w przypadku górnego akapitu), zostaje ona określona na podstawie szerokości obszaru zawierającego, pomniejszonej o szerokość marginesów, wypełnienia i obramowań. Jeśli natomiast szerokość jest podana jawnie (jak w przypadku drugiego akapitu), to zazwyczaj wszelkie różnice kompensowane są poprzez modyfikację szerokości prawego marginesu

Szerokość, marginesy oraz wartość `auto`

W przypadku większości elementów blokowych wartość `auto` właściwości `width` odpowiada szerokości obszaru zawierającego elementu, pomniejszonej o wypełnienie, obramowanie oraz marginesy. Obszar zawierający to szerokość przydzielona danemu elementowi przez jego element nadrzędny — często mylnie nazywana szerokością dziedziczną, choć absolutnie nie ma nic wspólnego z normalnymi zasadami dziedziczenia, obowiązującymi w CSS.

W przypadku elementów zawierających obrazki i inne obiekty (elementy `instantiate`) wartość `auto` szerokości odpowiada wewnętrznej szerokości elementu, czyli faktycznej szerokości obrazka lub obiektu zapisanego w pliku zewnętrznym. W przypadku elementów „plywających” wartości `auto` odpowiada szerokość wynosząca 0 pikseli. We wszystkich innych elementach wewnątrzstronowych właściwość `width` jest ignorowana.

W przypadku ręcznego określania właściwości `width`, `margin-left` oraz `margin-right`, gdy łączna szerokość wraz z wielkością obramowania i wypełnienia nie odpowiada szerokości obszaru zawierającego, jakąś szerokość trzeba będzie zmienić. I faktycznie, w takim przypadku przeglądarka zignoruje podaną wartość właściwości `margin-right` i przypisze jej wartość `auto` (rysunki 11.34 oraz 11.35). Jeśli ręcznie określisz wartość właściwości `width` i przypiszesz szerokości jednego z marginesów wartość `auto`, to margines ten będzie zmniejszany lub powiększany, tak by możliwe było skompensowanie różnic. Jeśli samodzielnie określisz wartość właściwości `width` i szerokości obu marginesów przypiszesz `auto`, to przeglądarka nada obu marginesom tę samą, maksymalną szerokość (co w efekcie spowoduje, że element zostanie wyśrodkowany).

Zauważ, że przeglądarka nigdy nie modyfikuje szerokości obramowania ani wypełnienia elementu.

Takie są zasady — niezależnie od tego, ile są wartości. Niestety, w przeglądarce IE 5 (oraz IE 6 działającej w trybie niestandardowym) określenie szerokości (właściwość `width`) jest interpretowane jako określenie sumy szerokości obszaru zawartości, wypełnienia, obramowania i marginesów, a nie — tak jak powinno być — samego obszaru zawartości. Z kolei w IE 6 marginesy oraz elementy zastępowane, umieszczane w elementach podrzędnych, mogą wpływać na szerokość bloku zawierającego, choć w dokumentacji jej twórcy zapewniają, że nic takiego nie ma prawa się dziać. Jak gdyby to wszystko i tak nie było dostatecznie skomplikowane!

Pozycjonowanie elementów w trzecim wymiarze

Względne oraz bezwzględne rozmieszczanie elementów może doprowadzić do sytuacji, gdy elementy będą się wzajemnie nakładać. W takiej sytuacji można określić, który element powinien być wyświetlony „powyżej” pozostałych.

Aby rozmieścić elementy w przestrzeni:

1. Wpisz `z-index`.
2. Wpisz `n`, gdzie `n` jest liczbą, określającą położenie elementu na „stosie” obiektów.

Wskazówki

- Im wyższa jest wartość właściwości `z-index`, tym wyżej element zostanie wyświetlony. Można wyobrazić sobie właściwość `z-index` jako określenie poziomu, na jakim element jest wyświetlany — przy czym użytkownicy, „szybując” nad stroną, patrzą na nią z góry, widząc najpierw te elementy, które zostały wyświetlone na najwyższym poziomie.
- Właściwości `z-index` można przypisywać zarówno wartości dodatnie, jak i ujemne.
- Jeśli jakieś elementy zostały umieszczone w elemencie o określonej wartości właściwości `z-index`, to wszystkie te elementy zostaną rozmieszczone w pierwszej kolejności zgodnie z wartościami zdefiniowanych w nich właściwości `z-index`, a dopiero potem — jako grupa elementów — zostaną rozmieszczone w szerszym kontekście strony. Wyobraźmy sobie na przykład, że w elemencie A właściwość `z-index` ma wartość 10, w elemencie B wartość 20, a w elemencie C wartość 30. Element B zawiera dwa elementy podrzędne: `bb`, w którym właściwość `z-index` ma wartość 35, oraz `bbb`, w którym właściwość ta ma wartość 5. Wszystkie te elementy zostaną wyświetlone w następującej kolejności (licząc od „góry”): C (który ma najwyższą wartość `z-index` na swoim poziomie strony), B (w którym `z-index` ma drugą co do wielkości wartość); następnie zostaną wyświetlone elementy podrzędne elementu B — `bb` oraz `bbb` (w takiej kolejności), a na samym końcu zostanie wyświetlony element A, którego właściwość `z-index` ma wartość mniejszą niż odpowiednia właściwość elementu B.
- Właściwość `z-index` nie jest dziedziczona.

```
body {background-color:#fff; color:#000;
padding:0 0 0 0;min-height:0 0 0 0}
img {position:absolute; top:250px; left:250;
background-color:black; top:0; left:0;
margin-left:-200px; z-index:1;}

img img {width:500px; height:500px;}

#content {position:absolute; top:0px; left:0px;
background-color:#fff;border-left:1px solid black;
padding:20px; z-index:2;}

#navigation {position:absolute; top:10px; left:250;
color:white; width:500px; z-index:3;}
#navigation a {display:block;color:#fff;
padding-left:10px; padding-right:10px;}
```

Rysunek 11.36. Wartości właściwości `z-index` elementów `content` oraz `navigation` można bez przeszkód zamieniać, o ile pozostaną one większe od wartości właściwości `z-index` elementu `bg`, wynoszącej 1 (wartość `z-index` w elemencie `content` musi być większa niż w elemencie `bg`, gdyż jego białe tło ukrywa część obrazka wychodzącą poza obszar, jaki ma zajmować element `bg`)



Rysunek 11.37. Teraz obszar nawigacyjny jest wyświetlony powyżej działu `bg`; w końcu wyjaśniło się, dlaczego łącza zostały wyświetlone na białym

```

#calendar {background-color:#f9f9; color:white; padding:5px;
margin-left:5px; margin-top:10px; font-size:.8em;
width:150px;}

#calendar ul {margin:0;color:white;}

```

Rysunek 11.38. W pierwszej kolejności sformatowałam element `div` o identyfikatorze `calendar`, tak aby ładnie wyglądał



Rysunek 11.39. Kalendarz wygląda całkiem ładnie u dołu strony, ale według mnie jest zbyt długi

```

#dates {width:150px;height:2.5em;overflow:auto}

```

Rysunek 11.40. A zatem jawnie określiłam wysokość (`height`) oraz szerokość (`width`) akapitu `dates` umieszczonego wewnątrz elementu `calendar`, dzięki czemu wiem, jakie wymiary będzie miał wynikowy akapit. Następnie dodałam właściwość `overflow`, aby odpowiednio ułożyć zawartość akapitu, która nie mieści się w jego prostokącie



Rysunek 11.41. Teraz akapit `dates` ma dokładnie 150 pikseli szerokości i 2 em wysokości; są w nim wyświetlane paski przewijania, aby użytkownicy mogli zobaczyć całą jego zawartość

Określanie sposobu wyświetlania zawartości elementu

Elementy nie zawsze są wyświetlane wewnątrz swych prostokątów. Czasami prostokąty te po prostu nie są odpowiednio duże. Może się także zdarzyć, że zawartość została umieszczona poza prostokątem danego elementu. Niezależnie od przyczyny, za pomocą właściwości `overflow` można kontrolować, co się będzie działo z zawartością znajdującą się poza prostokątem elementu.

Aby określić sposób wyświetlania zawartości elementu

1. Wpisz `overflow`.
2. Wpisz `visible`, aby nadać elementowi większy rozmiar, wystarczający do wyświetlenia całej jego zawartości (to domyślna wartość właściwości `overflow`),
lub wpisz `hidden`, aby ukryć zawartość, która nie mieści się w prostokącie elementu,
lub wpisz `scroll`, aby w elemencie zawsze były wyświetlane paski przewijania, umożliwiające użytkownikom wyświetlenie zawartości nie mieszczącej się w prostokącie elementu,
lub wpisz `auto`, aby paski przewijania były wyświetlane wyłącznie w razie konieczności.

Wskazówki

- Pamiętaj, że w przeglądarce Internet Explorer 6 zakładane jest, że nie wiesz, co robisz, nadając elementowi podrzędnemu większe wymiary niż ma element nadrzędny, i powiększa element nadrzędny tak, aby element podrzędny w nim się zmieścił. Jedyнным wyjątkiem od tej reguły jest sytuacja, w której właściwości `overflow` zostanie przypisana dowolna wartość oprócz `visible` (domyślnej). W takim przypadku elementowi nadrzędnemu zostanie nadana zamierzona wielkość, a właściwość `overflow` będzie mogła spełnić swoje zadanie.
- Domyślną wartością właściwości `overflow` jest `visible`. Właściwość ta nie jest dziedziczona.

Otaczanie elementów tekstem

Nic nie stoi na przeszkodzie, aby otoczyć element tekstem (bądź także innymi elementami). Niestety przeglądarki czasami nieprawidłowo obsługują właściwość `float`.

Aby otoczyć element tekstem:

1. Wpisz `float`.
2. Wpisz `left`, jeśli chcesz, aby element znajdował się z lewej strony, a tekst otaczał go z prawej, lub wpisz `right`, jeśli chcesz, aby element znajdował się z prawej strony, a tekst otaczał go z lewej.
3. Skorzystaj z właściwości `width` (patrz strona 190), aby jawnie określić szerokość elementu.

Wskazówki

- Pamiętaj, że kierunek wybrany w definicji właściwości dotyczy położenia elementu, a nie tekstu. Definicja `float: left` spowoduje umieszczenie tekstu z prawej strony elementu.
- Sztuczka pozwalająca na rozmieszczenie tekstu pomiędzy elementami polega na umieszczeniu elementu *bezpośrednio przed* tekstem, który ma go otaczać.
- Informacje na temat staromodnego, odrzuconego, ale powszechnie obsługiwanego sposobu wyświetlania tekstu pomiędzy obrazkami można znaleźć na stronie 110, w sekcji „Otaczanie obrazów tekstem”.
- Jeśli w więcej niż jednym elemencie podano tę samą wartość właściwości `float`, to pierwszy element jest umieszczony najdalej we wskazanym kierunku, drugi jest umieszczany za pierwszym i tak dalej (rysunki 11.44 oraz 11.45).
- Elementy, które nie są zastępowane i w których nie została określona wartość właściwości `width`, nie będą poprawnie otaczane.
- Właściwość `float` nie jest dziedziczona.

```
#calendar {position: absolute; top: 0px; left: 0px;
background-color: #fff; border-left: 3px solid black;
padding: 10px; z-index: 1;}

#content {background-color: #f0f0f0; color: white; padding: 5px;
margin-left: 5px; margin-top: 7px; float: right; width: 150px; float: right;}
```

Rysunek 11.42. Chcemy, aby cały dział `calendar` strony był dosunięty do prawej krawędzi strony i otaczany tekstem



Rysunek 11.43. Tekst umieszczony bezpośrednio za działem `calendar` otacza go z lewej strony

```
<div>Zamieszaj</div>

<div id="calendar">
```

Rysunek 11.44. Teraz wyobraźmy sobie, że przed elementem `calendar` znajduje się obrazek, w którym właściwości `float` także przypisano wartość `right`



Rysunek 11.45. Ponieważ niewielki obrazek jest umieszczony w kodzie (X)HTML przed elementem `div`, zostanie on wyświetlony z jego prawej strony

```

#calendar {position:absolute; top: 5px; left: 30%;
background-color:#fff;border-left: 1px solid black;
padding: 20px; z-index: 2;}

#calendar #calendar {background:#339; color:white; padding:5px;
margin-left:5px; margin-top:7px; font-size:.8em;
width:150px; float:right; clear:right;}

#calendar #t {margin:0;color:white;}

.darce {width:150px;height:1.5em;overflow:auto;}

.icon {margin-top:7px;float:right;}

```

Rysunek 11.46. Właściwość `clear` jest umieszczana w elementach, które nie mają być wyświetlane obok innych „otaczanych” elementów. W tym przykładzie chcemy, aby żadne inne elementy nie znajdowały się z prawej strony działu `calendar` i dlatego w dotyczącej go regule arkusza stylów umieściłam właściwość `clear: right`



Rysunek 11.47. Dział `calendar` nie zostanie wyświetlony aż do momentu, gdy z jego prawej strony nie będzie już żadnych innych elementów (czyli przeglądarka wyświetli go dopiero poniżej niewielkiego obrazka)

Kontrola sposobu otaczania elementów

Można określić, jakie elementy mogą być otaczane przez inne elementy, a jakie nie. Aby element nie był wyświetlany obok innego elementu, obok którego nie powinien być umieszczany, należy posłużyć się właściwością `clear`.

Aby kontrolować sposób otaczania elementów:

1. Wpisz `clear:`
2. Wpisz `left`, aby inne elementy nie otaczały bieżącego elementu z lewej strony,
lub wpisz `right`, aby inne elementy nie otaczały bieżącego elementu z prawej strony,
lub wpisz `both`, aby dany element nie był otaczany ani z prawej, ani z lewej strony,
lub wpisz `none`, aby inne elementy nie mogły otaczać bieżącego elementu ani z lewej, ani z prawej strony.

Wskazówki

- Właściwość `clear` sprawia, że element, w którym została ona zdefiniowana, nie będzie wyświetlany aż do momentu, gdy po podanej stronie nie będą już wyświetlane żadne inne elementy.
- Właściwość `clear` jest dodawana do elementów, obok których nie mają być wyświetlane żadne inne „otaczane” elementy. A zatem jeśli chcesz, aby dany element nie był wyświetlany aż do momentu, gdy z jego prawej strony nie będą już umieszczone żadne inne „otaczane” elementy, zdefiniuj w nim (a nie w tych innych elementach) właściwość `clear:right`.
- Sposób użycia właściwości `clear` przypomina posługiwanie się znacznikiem `br` z (odrzuconym) atrybutem `clear` (patrz strona 112).

Wyrównywanie elementów w pionie

Aby poprawić atrakcyjność strony, elementy można wyrównywać na wiele różnych sposobów.

Aby wyrównać elementy w pionie:

1. Wpisz `vertical-align:`.
2. Wpisz `baseline`, aby wyrównać elementy do podstawy (ang.: *baseline*) elementu nadrzędnego, lub wpisz `middle`, aby wyrównać środki elementów do środka elementu nadrzędnego, lub wpisz `sub`, aby umieścić element w dolnym indeksie elementu nadrzędnego, lub wpisz `super`, aby umieścić element górnym indeksie elementu nadrzędnego, lub wpisz `text-top`, aby wyrównać górną część elementu z górną częścią elementu nadrzędnego, lub wpisz `text-bottom`, aby wyrównać dolną część elementu z dolną częścią elementu nadrzędnego, lub wpisz `top`, aby wyrównać górną część elementu z górną częścią najwyższego elementu w linii, lub wpisz `bottom`, aby wyrównać dolną część elementu z dolną częścią najniższego elementu w linii, lub podaj procent wysokości elementu (możesz użyć wartości dodatniej lub ujemnej).

```
body {background-color:#fff; color:#000;
padding:0 0 10px; margin:0 0 0 0}
#bg {position: absolute; top: 20px; left: 20px;
background-color:black; top:0; left:0;
margin-left:-200px; z-index: 1;}
#bg img { width:550px; height:650px; }

#content {position: absolute; top: 0px; left: 30%;
background-color:#fff; border-left: 1px solid black;
padding: 20px; z-index: 2;}
img.line {vertical-align: top;}
```

Rysunek 11.48. Jedynie w obrazkach należących do klasy `line` użyliśmy wyrównania elementu do górnej krawędzi linii (`vertical-align: top`)



Rysunek 11.49. Wszystkie trzy obrazki należące do klasy `line` są wyrównane w pionie względem swych górnych krawędzi