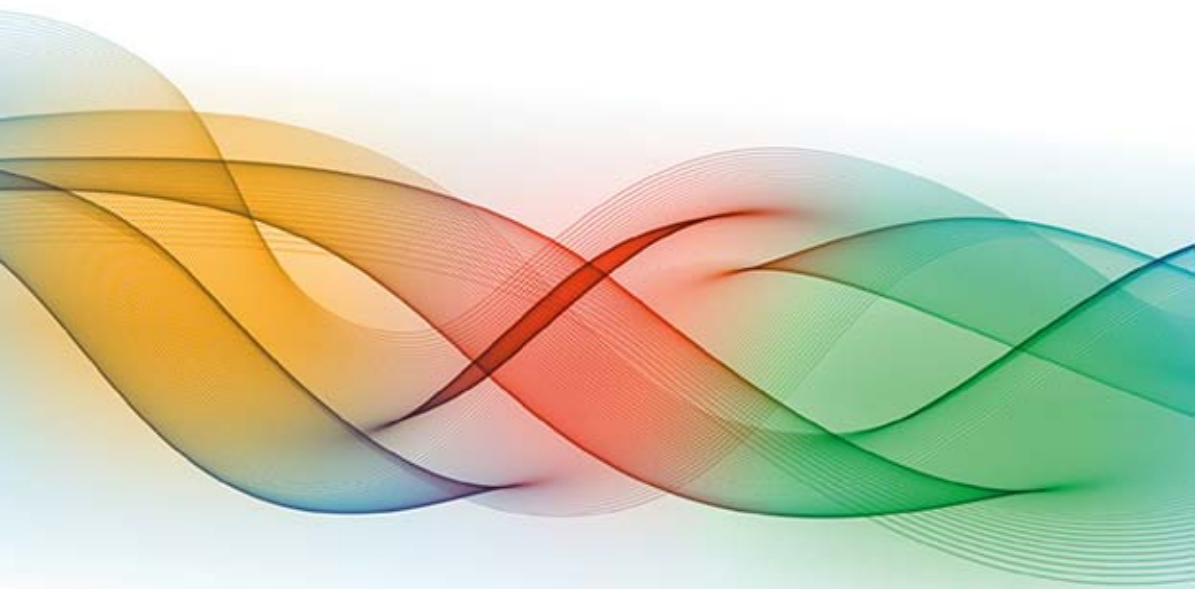


Dawid Borycki

JavaScript i jQuery

Kompletny przewodnik dla programistów
interaktywnych aplikacji internetowych
w Visual Studio



JavaScript i jQuery — fantastyczny duet w każdej aplikacji!

Poznaj technologię JavaScript i wykorzystaj jej kolosalne możliwości
Ułatw sobie pracę za pomocą elastycznych narzędzi z biblioteki jQuery i jQuery UI

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

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

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

Redaktor prowadzący: Ewelina Burska

Projekt okładki: Studio Gravite/Olsztyn

Obarek, Pokoński, Pazdrijowski, Zaprucki

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock.

Wydawnictwo HELION

ul. Kościuszki 1c, 44-100 GLIWICE

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

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

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

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

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-246-8283-6

Copyright © Helion 2014

Printed in Poland.

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

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

Spis treści

Przedmowa	7
Wstęp	9
Część I Technologia JavaScript	11
Rozdział 1. Wprowadzenie do JavaScript	13
Osadzenie skryptu	15
Zmienne	16
Funkcje	18
Funkcje a zasięg zmiennych	21
Instrukcje sterujące i operatory logiczne	24
Warunki a stałe	25
Warunki a zmienne	26
Zdania logiczne zawierające zmienne i stałe	28
Koniunkcja i alternatywa kryteriów	28
Instrukcja switch	32
Operatory inkrementacji, dekrementacji oraz przypisania	34
Pętle	35
Polecenie while	35
Konstrukcja do while	37
Pętla for	38
Instrukcje break i continue	41
Podsumowanie	43
Rozdział 2. Konfiguracja obsługi JavaScript i biblioteki skryptów	45
Problem braku obsługi JavaScript	45
Konfiguracja obsługi JavaScript w przeglądarce Internet Explorer	46
Wyłączenie obsługi JavaScript w przeglądarce Mozilla Firefox	47
Konfiguracja przeglądarki Google Chrome do obsługi JavaScript	48
Biblioteki skryptów	49
Podsumowanie	52
Rozdział 3. Debugowanie kodu JavaScript	53
Wprowadzenie	53
Narzędzia deweloperskie w Internet Explorer 9	53
Konsola	55
Czujka	57

Rozpoczęcie debugowania i punkty przzerwania	58
Zmienne lokalne	60
Stos wywołań	61
Inspekcja kodu JavaScript w przeglądarce Mozilla Firefox	61
Konsola WWW	61
Brudnopis	63
Podsumowanie	64
Rozdział 4. Dostęp do wybranych elementów stron internetowych	65
Wprowadzenie	65
Pole tekstowe	66
Modyfikacja zawartości wybranego elementu strony	67
Właściwości obiektów HTML	69
Modyfikacja właściwości wybranego obiektu HTML	69
Modyfikacja stylu elementu strony	73
Podsumowanie	78
Rozdział 5. Właściwości okna przeglądarki	79
Wprowadzenie	79
Dynamiczna kontrola nowego okna przeglądarki internetowej	79
Przenoszenie i dynamiczna zmiana rozmiarów okna przeglądarki	84
Podsumowanie	88
Rozdział 6. Obsługa zdarzeń i drzewo DOM	89
Wprowadzenie	89
Obsługa myszy	89
Obsługa klawiatury oraz metody zdarzeniowe towarzyszące ładowaniu elementów HTML	94
Podsumowanie	103
Rozdział 7. Tabele danych, źródła XML oraz pętla for in	105
Wprowadzenie	105
Odczytanie zawartości pliku XML	106
Pobieranie wybranych informacji z pliku XML	108
Tworzenie tabeli danych	110
Pętla for in	113
Podsumowanie	114
Rozdział 8. Formularze	115
Wprowadzenie	115
Walidacja danych formularza	116
Formularze a wyrażenia regularne	120
Zdarzenie onchange	122
Formatowanie błędnie wypełnionych pól	123
Potwierdzenie wysłania i wyczyszczenia zawartości pól formularza	125
Podsumowanie	126
Rozdział 9. Obiekty wbudowane	127
Wprowadzenie	127
Math	127
Date	129
String	131
JSON	133
Konstruowanie i obsługa własnych obiektów	135

Właściwości przeglądarki	138
Navigator	138
History	139
Location	140
Podsumowanie	141
Rozdział 10. Animacje	143
Wprowadzenie	143
Ściemnianie i rozjaśnianie	143
Zmiana rozmiaru	147
Ruch	151
Podsumowanie	155
Część II Biblioteka jQuery	157
Rozdział 11. Podstawy jQuery	159
Wprowadzenie i dyskretny JavaScript	159
Importowanie biblioteki jQuery i zdarzenie \$(document).ready	160
Selektory, czyli dostęp do elementów drzewa DOM	162
Konstruowanie i wykorzystanie selektorów	164
Filtrowanie atrybutów i dodatkowe selektory	171
Uzupełnienia	175
Manipulacja drzewem DOM	176
Zdarzenia	178
Rejestracja zdarzenia	179
Wyłączenie obsługi zdarzenia	180
Ręczne wyzwolenie zdarzenia	181
Zablokowanie domyślnej obsługi zdarzenia	183
Argumenty zdarzenia	184
Funkcje zwrotne, czyli przetwarzanie asynchroniczne	185
Podsumowanie	188
Rozdział 12. Zaawansowane aspekty biblioteki jQuery	189
Animacje	189
Formularze	195
AJAX	198
Wtyczki i rozszerzenia	204
Podsumowanie	208
Rozdział 13. jQuery UI	209
Wprowadzenie i przygotowanie środowiska pracy	209
Wtyczki	210
ProgressBar	210
Slider	212
Datepicker	214
Dialog	219
Button	226
Tabs	232
Accordion	238
Interakcja z komponentami	241
Przenoszenie	241
Upuszczanie	246
Zaznaczanie	250

Sortowanie	252
Zmiana rozmiaru	256
Efekty animacji	258
Przełączanie klas CSS	258
Dodawanie, usuwanie i zmiana klasy CSS	260
Animowanie kolorów	261
Wbudowane efekty animacji	263
Easing	266
Tworzenie własnych motywów	269
Podsumowanie	270
Skorowidz	271

Rozdział 10.

Animacje

Wprowadzenie

JavaScript w połączeniu z kaskadowymi arkuszami stylów pozwala uatrakcyjnić projektowaną witrynę efektami animacji. Dzięki temu poprawi się interaktywność całej aplikacji internetowej, która ułatwi użytkownikowi jej obsługę.

W praktyce implementowanie animacji elementów drzewa DOM sprowadza się do umiejętnego modyfikowania właściwości style wybranego obiektu HTML. Ponieważ preferencje estetyczne są kwestią indywidualnego wyboru, więc w tym podrozdziale pokażę, w jaki sposób wykonać proste efekty animacji, takie jak: animacje wejścia i wyjścia (rozjaśnianie i ściemnianie) obiektów, zmiana ich rozmiarów i położenia. Moim celem nie będzie utworzenie pięknych animacji, a jedynie przedstawienie podstaw niezbędnych do samodzielnego implementowania własnych animacji.

Ściemnianie i rozjaśnianie

Dynamiczna kontrola poziomu jasności wybranego elementu HTML sprowadza się do modyfikacji właściwości `opacity`, którą w zależności od przeglądarki internetowej można zmodyfikować za pomocą właściwości `style.filter` w przypadku Internet Explorera lub właściwości `style.opacity` w przypadku pozostałych przeglądarek.

Zanim przejdę do zaimplementowania funkcji, której celem będzie animacja wybranego elementu HTML poprzez ciągłą zmianę poziomu jego jasności, pokażę, w jaki sposób zmienić jego właściwość `opacity`. W tym celu:

1. Utwórz nową witrynę HTML o nazwie *Animacje.htm*.
2. W sekcji `<title>` witryny wpisz jej nazwę, czyli Animacje.

3. Korzystając z listingu 10.1, uzupełnij nagłówek strony *Animacje* o definicję klas kaskadowych arkuszy stylów dla bloku `div`.

Listing 10.1. Definicja formatowania bloku `div`

```
<style type="text/css">
  div.zolty
  {
    color: #996600;
    font-weight: bold;
    background-color: #FFFFCC;
    border-color: #996600;
    border-style: solid;
    border-width: 2px;
    width: 100px;
    text-align: center;
  }

  div.brazowy
  {
    color: #FFFFCC;
    font-weight: bold;
    background-color: #996600;
    border-color: #FFFFCC;
    border-style: solid;
    border-width: 1px;
    width: 100px;
    text-align: center;
  }
</style>
```

4. W nagłówku witryny *Animacje* umieść polecenia z listingu 10.2.

Listing 10.2. Kontrola poziomu jasności wybranego elementu HTML za pomocą JavaScript

```
function ustawJasnoc(idElementu, poziomJasnosci) {
  var element = document.getElementById(idElementu);

  if(!element) return;

  element.style.opacity = poziomJasnosci / 10;
  element.style.filter = 'alpha(opacity=' + poziomJasnosci * 10 + ')';
}

function odczytajJasnoc(idElementu) {
  var element = document.getElementById(idElementu);

  if(!element) return;

  var jasnoc;
  if (element.style.filter) {
    var indeksP = element.style.filter.indexOf('=');
    var indeksK = element.style.filter.indexOf(')');

    jasnoc = parseFloat(element.style.filter.substring(indeksP + 1,
      ↪indeksK)) / 10;
  }
}
```



```
else
    jasnoc = element.style.opacity * 10;

return jasnoc;
}
```

5. Sekcję `<body>` witryny *Animacje* zdefiniuj według wzoru z listingu 10.3.

Listing 10.3. Zdarzenia *onmouseover* oraz *onmouseout* a poziom jasności bloku *div*

```
<body onload="ustawJasnoc('blok1', 10);">
  <div id="blok1" class="zolty" onmouseover="ustawJasnoc('blok1', 5);"
    ↪onmouseout="ustawJasnoc('blok1', 10)">Wydawnictwo Helion</div>
</body>
```

Dwa aspekty powyższego rozwiązania zasługują na szczególną uwagę. Są nimi funkcje *ustawJasnoc* oraz *odczytajJasnoc*. Pierwsza z nich pozwala zmodyfikować poziom jasności wybranego fragmentu witryny internetowej poprzez kontrolę właściwości *opacity*. Funkcja ta jest kompatybilna z różnymi przeglądarkami i pozwala ustawić jasność obiektów HTML na wartość z zakresu od 0, dla której dany obiekt będzie całkowicie przezroczysty (niewidoczny), do 10, dla której dany obiekt nie będzie przezroczysty.

W zależności od rodzaju przeglądarki internetowej właściwość *opacity* można zmodyfikować za pomocą pola *style.opacity*, do którego należy po prostu przypisać odpowiednią wartość, reprezentującą poziom przezroczystości obiektu HTML. Właściwość *style.opacity* może przyjmować wartości z zakresu od 0 do 1. Z tego powodu wartość parametru przekazanego do funkcji *ustawJasnoc* jest dzielona przez współczynnik równy 10. W przypadku przeglądarki Internet Explorer przezroczystość obiektu HTML można zmodyfikować za pomocą właściwości *style.filter*, do której należy przypisać łańcuch tekstowy w postaci *"alpha(opacity=jasność)"*, gdzie parametr jasność jest wyrażony w procentach i odpowiada procentowemu poziomowi przezroczystości danego elementu HTML.

Ze względu na fakt, że sposób kontroli jasności (przezroczystości) obiektów HTML zależy od wykorzystywanej przeglądarki internetowej, również i odczytanie poziomu przezroczystości danego obiektu jest uzależnione od rodzaju przeglądarki. Wobec tego funkcja *odczytajJasnoc* z listingu 10.2 najpierw weryfikuje, czy własność *filter* obiektu *style* została wcześniej zdefiniowana. Jeśli tak, to pobiera wartość jasności znajdującą się pomiędzy znakami *=* oraz *)*. W przeciwnym wypadku odczytuje własność *style.opacity*.

Po otwarciu witryny *Animacje.htm* w domyślnej przeglądarce internetowej i przesunięciu kursora myszy w obrębie napisu *Wydawnictwo Helion* nastąpi jego ściemnienie. Po usunięciu kursora myszy z regionu wyznaczonego przez ten napis poziom jego przezroczystości zostanie przywrócony do 100%.

W kolejnym przykładzie zmodyfikuję kod witryny *Animacje.htm* w taki sposób, aby zmiana jasności napisu *Wydawnictwo Helion* realizowana była w sposób ciągły. Wymaga to dokonania następujących czynności:

1. Skrypt witryny Animacje.htm uzupełnij o polecenia z listingu 10.4.

Listing 10.4. Animowanie zmian jasności obiektu HTML

```

var animacjaZmianyJasnosci;

function animujJasnoc(idElementu, ściemnianie, jasnocDocelowa,
↳krokZmianyJasnosci) {
    var element = document.getElementById(idElementu);

    if (!element) return;

    var aktualnaJasnocElementu = odczytajJasnoc(idElementu);
    var nowaJasnoc;

    if (ściemnianie) {
        nowaJasnoc = aktualnaJasnocElementu - krokZmianyJasnosci;
        if (nowaJasnoc < jasnocDocelowa) nowaJasnoc = jasnocDocelowa;
    }
    else {
        nowaJasnoc = aktualnaJasnocElementu + krokZmianyJasnosci;
        if (nowaJasnoc > jasnocDocelowa) nowaJasnoc = jasnocDocelowa;
    }

    ustawJasnoc(idElementu, nowaJasnoc);

    // Czy docelowy poziom jasności został już osiągnięty?
    if (nowaJasnoc != jasnocDocelowa) {
        // Jeśli nie, to animuj dalej...

        animacjaZmianyJasnosci = setTimeout("animujJasnoc('" + idElementu
            + "', " + ściemnianie
            + ", " + jasnocDocelowa
            + ", " + krokZmianyJasnosci
            + ");", 50);
    }
    else {
        // W przeciwnym wypadku przerwij animację
        zatrzymajAnimacje(animacjaZmianyJasnosci);
    }
}

function zatrzymajAnimacje(animacja) {
    clearTimeout(animacja);
}

```

2. Definicję sekcji <body> witryny zmodyfikuj według następującego wzoru:

```

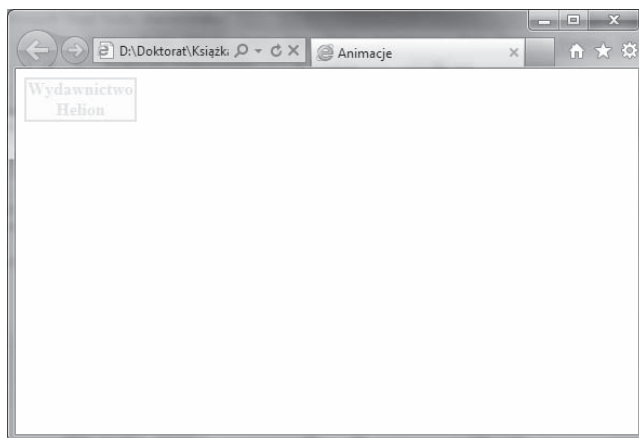
<body onload="ustawJasnoc('blok1', 10);">
<div id="blok1" class="zolty" onmouseover="ustawJasnoc('blok1', 5);"
↳onmouseout="ustawJasnoc('blok1', 10)">Wydawnictwo Helion</div>
<div id="blok1" class="zolty" onmouseover="zatrzymajAnimacje(
↳animacjaZmianyJasnosci);animujJasnoc('blok1', true, 2, 1);"
↳onmouseout="zatrzymajAnimacje(animacjaZmianyJasnosci);animujJasnoc(
↳'blok1', false, 10, 1);">Wydawnictwo Helion</div>
</body>

```

Po zaimplementowaniu powyższych zmian w witrynie *Animacje.htm* należy ją otworzyć w domyślnej przeglądarce internetowej. Umieszczenie kursora myszy ponad napisem *Wydawnictwo Helion* spowoduje wyzwolenie animacji zmieniającej jego jasność. Efekt końcowy powinien być analogiczny do przedstawionego na rysunku 10.1. Przesunięcie kursora myszy w obszarze zajmowanym przez napis *Wydawnictwo Helion* powoduje wyzwolenie kodu JavaScript związanego ze zdarzeniem *mouseover*, czyli wywołanie metod *zatrzymajAnimacje* oraz *animujJasnoc*. Pierwsza z nich powoduje zatrzymanie wcześniej wykonywanej animacji i rozpoczęcie nowej. Ciągła zmiana poziomu jasności wybranego elementu HTML została zaimplementowana w funkcji *animujJasnoc*, której działanie sprowadza się do odczytania bieżącego poziomu jasności danego obiektu HTML. W następnym kroku, w zależności od wartości parametru *ściemnianie*, zmniejszam lub odpowiednio zwiększam nowy poziom jasności. Jeśli ten nowy poziom odpowiada wartości docelowej, to animacja zostaje zakończona. W przeciwnym wypadku następuje ponowne wywołanie funkcji *animujJasnoc* przy użyciu znanego już mechanizmu, wykorzystującego funkcję *setTimeout*.

Rysunek 10.1.

Efekt końcowy animacji typu ściemnianie



Dzięki temu, że funkcja *animujJasnoc* jest napisana stosunkowo ogólnie, można ją również wykorzystać zarówno do ściemniania, jak i rozjaśniania wybranego obiektu HTML. Z tego powodu funkcję *animujJasnoc* skojarzyłem ze zdarzeniem *mouseout*, z tą tylko różnicą, że w przypadku rozjaśniania obiektu poziomem docelowym jasności jest jasność maksymalna, czyli 10. Wobec tego usunięcie kursora myszy z obszaru wyznaczonego przez napis *Wydawnictwo Helion* spowoduje jego rozjaśnienie.

Zmiana rozmiaru

W tym podrozdziale zaimplementuję animację zmiany rozmiaru wybranego elementu HTML. Realizację tego zadania można wykonać poprzez analogię do przykładu przedstawionego w poprzednim podrozdziale. Wobec tego wykonaj następujące kroki:

1. Sekcję `<body>` witryny *Animacje.htm* uzupełnij o następujące polecenia:

```
<div id="blok2" class="brazowy" onmouseover="animujSzerokosc('blok2', true,
↳100, 125, 1);" onmouseout="animujSzerokosc('blok2', false,
↳parseInt(this.style.width), 100, 1);">Wydawnictwo Helion</div>
```

2. W nagłówku witryny przejdź do sekcji `<script>` i umieść w niej definicję funkcji z listingu 10.5.

Listing 10.5. *Animacja zmiany szerokości wybranego elementu HTML*

```
function animujSzerokosc(idElementu, rozszerzanie, szerokoscPocatkowa,
↳szerokoscDocelowa, krokZmianySzerokosci) {
    var element = document.getElementById(idElementu);

    if (!element) return;

    var aktualnaSzerokoscElementu =
        element.style.width ? parseInt(element.style.width) : szerokoscPocatkowa;
    var nowaSzerokosc;

    if (rozszerzanie) {
        nowaSzerokosc = aktualnaSzerokoscElementu + krokZmianySzerokosci;
        if (nowaSzerokosc > szerokoscDocelowa) nowaSzerokosc = szerokoscDocelowa;
    }
    else {
        nowaSzerokosc = aktualnaSzerokoscElementu - krokZmianySzerokosci;
        if (nowaSzerokosc < szerokoscDocelowa) nowaSzerokosc = szerokoscDocelowa;
    }

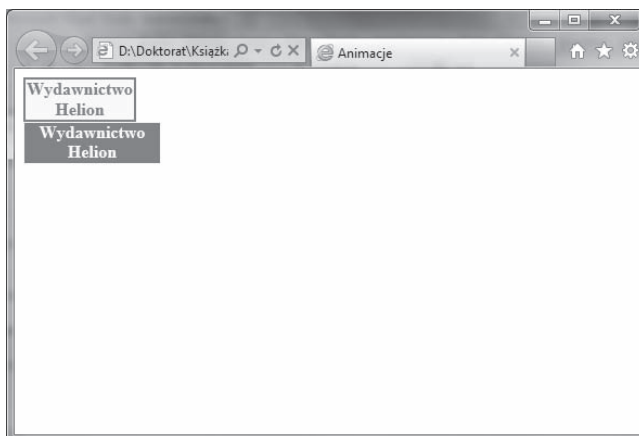
    element.style.width = nowaSzerokosc + "px";

    // Czy docelowa szerokość została już osiągnięta?
    if (nowaSzerokosc != szerokoscDocelowa) {
        // Jeśli nie, to animuj dalej...
        element.animacjaZmianySzerokosci = setTimeout("animujSzerokosc('" +
↳idElementu
        + "', " + rozszerzanie
        + ", " + szerokoscPocatkowa
        + ", " + szerokoscDocelowa
        + ", " + krokZmianySzerokosci
        + ");", 10);
    }
    else {
        // W przeciwnym wypadku przerwij animację
        zatrzymajAnimacje(element.animacjaZmianySzerokosci);
    }
}
```

Konstrukcja powyższego przykładu jest bardzo podobna do animacji przezroczystości obiektów HTML. Jednakże zamiast właściwości `style.opacity` modyfikuję wartość zapisaną w polu `style.width`. Dzięki temu po umieszczeniu kursora myszy w obrębie wyznaczonym przez napis *Wydawnictwo Helion* uzyskuje się efekt analogiczny do przedstawionego na rysunku 10.2.

Rysunek 10.2.

*Efekt końcowy
animacji zmiany
szerokości
elementu HTML*



Animacja zmiany rozmiarów elementów HTML znajduje zastosowanie w wyróżnianiu aktywnych (aktualnie zaznaczonych) elementów menu aplikacji internetowej. W celu zasymulowania takiego menu trzykrotnie powiele bieżący blok `div` zawierający napis *Wydawnictwo Helion*. Po tej operacji sekcja `<body>` witryny *Animacje.htm* powinna mieć następującą postać:

```
<body onload="ustawJasnosci('blok1', 10);">
  <!--<div id="blok1" class="zolty" onmouseover="ustawJasnosci('blok1', 5);"
  ↳onmouseout="ustawJasnosci('blok1', 10)">Wydawnictwo Helion</div-->
  <div id="blok1" class="zolty" onmouseover="zatrzymajAnimacje(
  ↳animacjaZmianyJasnosci);animujJasnosci('blok1', true, 2, 1);"
  ↳onmouseout="zatrzymajAnimacje(animacjaZmianyJasnosci);animujJasnosci('blok1',
  ↳false, 10, 1);">Wydawnictwo Helion</div>
  <div id="blok2" class="brazowy" onmouseover="zatrzymajAnimacje(
  ↳animacjaZmianySzerokosci);animujSzerokosc('blok2', true, 100, 125, 1);"
  ↳onmouseout="zatrzymajAnimacje(animacjaZmianySzerokosci);animujSzerokosc(
  ↳'blok2', false, parseInt(this.style.width), 100, 1);">Wydawnictwo
  ↳Helion</div>
  <div id="blok3" class="brazowy" onmouseover="zatrzymajAnimacje(
  ↳animacjaZmianySzerokosci);animujSzerokosc('blok3', true, 100, 125, 1);"
  ↳onmouseout="zatrzymajAnimacje(animacjaZmianySzerokosci);animujSzerokosc(
  ↳'blok3', false, parseInt(this.style.width), 100, 1);">Wydawnictwo
  ↳Helion</div>
  <div id="blok4" class="brazowy" onmouseover="zatrzymajAnimacje(
  ↳animacjaZmianySzerokosci);animujSzerokosc('blok4', true, 100, 125, 1);"
  ↳onmouseout="zatrzymajAnimacje(animacjaZmianySzerokosci);animujSzerokosc(
  ↳'blok4', false, parseInt(this.style.width), 100, 1);">Wydawnictwo
  ↳Helion</div>
  <div id="blok5" class="brazowy" onmouseover="zatrzymajAnimacje(
  ↳animacjaZmianySzerokosci);animujSzerokosc('blok5', true, 100, 125, 1);"
  ↳onmouseout="zatrzymajAnimacje(animacjaZmianySzerokosci);animujSzerokosc(
  ↳'blok5', false, parseInt(this.style.width), 100, 1);">Wydawnictwo
  ↳Helion</div>
</body>
```

Po otwarciu tak zmodyfikowanej witryny w domyślnej przeglądarce internetowej i przesuwaniu kursora myszy po bieżących elementach HTML (rysunek 10.2) nietrudno stwierdzić, że efekt działania aplikacji jest daleki od zamierzonego. Poszczególne efekty

animacji są zatrzymywane i uruchamiane w sposób chaotyczny. Dzieje się tak dlatego, że każdorazowe przesunięcie kursora myszy z jednego obiektu HTML na inny zatrzymuje animację danego elementu i uruchamia animację kolejnego obiektu HTML. Powinniśmy zapewnić, aby efekt animacji wszystkich elementów wykonywany był niezależnie. Wobec tego funkcję `animujSzerokosc` należy zmodyfikować według wzoru z listingu 10.6. Natomiast w sekcji `<body>` witryny *Animacje.htm* należy usunąć wywołania funkcji `zatrzymajAnimacje`:

```
<body onload="ustawJasnoc('blok1', 10);">
  <!--<div id="blok1" class="zolty" onmouseover="ustawJasnoc('blok1', 5);"
    ↳onmouseout="ustawJasnoc('blok1', 10)">Wydawnictwo Helion</div>-->
  <div id="blok1" class="zolty" onmouseover="zatrzymajAnimacje(
    ↳animacjaZmianyJasnosci);animujJasnoc('blok1', true, 2, 1);"
    ↳onmouseout="zatrzymajAnimacje(animacjaZmianyJasnosci);animujJasnoc('blok1',
    ↳false, 10, 1);">Wydawnictwo Helion</div>
  <div id="blok2" class="brazowy" onmouseover="animujSzerokosc('blok2', true,
    ↳100, 125, 1);" onmouseout="animujSzerokosc('blok2', false,
    ↳parseInt(this.style.width), 100, 1);">Wydawnictwo Helion</div>
  <div id="blok3" class="brazowy" onmouseover="animujSzerokosc('blok3', true,
    ↳100, 125, 1);" onmouseout="animujSzerokosc('blok3', false,
    ↳parseInt(this.style.width), 100, 1);">Wydawnictwo Helion</div>
  <div id="blok4" class="brazowy" onmouseover="animujSzerokosc('blok4', true,
    ↳100, 125, 1);" onmouseout="animujSzerokosc('blok4', false,
    ↳parseInt(this.style.width), 100, 1);">Wydawnictwo Helion</div>
  <div id="blok5" class="brazowy" onmouseover="animujSzerokosc('blok5', true,
    ↳100, 125, 1);" onmouseout="animujSzerokosc('blok5', false,
    ↳parseInt(this.style.width), 100, 1);">Wydawnictwo Helion</div>
</body>
```

Listing 10.6. Dynamiczna modyfikacja definicji obiektu

```
var animacjaZmianySzerokosci;

function animujSzerokosc(idElementu, rozszerzanie, szerokoscPocztkowa,
↳szerokoscDocelowa, krokZmianySzerokosci) {
  var element = document.getElementById(idElementu);

  if (!element) return;

  if (element.animacjaZmianySzerokosci)
    zatrzymajAnimacje(element.animacjaZmianySzerokosci);

  var aktualnaSzerokoscElementu =
    element.style.width ? parseInt(element.style.width) : szerokoscPocztkowa;
  var nowaSzerokosc;

  if (rozszerzanie) {
    nowaSzerokosc = aktualnaSzerokoscElementu + krokZmianySzerokosci;
    if (nowaSzerokosc > szerokoscDocelowa) nowaSzerokosc = szerokoscDocelowa;
  }
  else {
    nowaSzerokosc = aktualnaSzerokoscElementu - krokZmianySzerokosci;
    if (nowaSzerokosc < szerokoscDocelowa) nowaSzerokosc = szerokoscDocelowa;
  }

  element.style.width = nowaSzerokosc + "px";
}
```

```

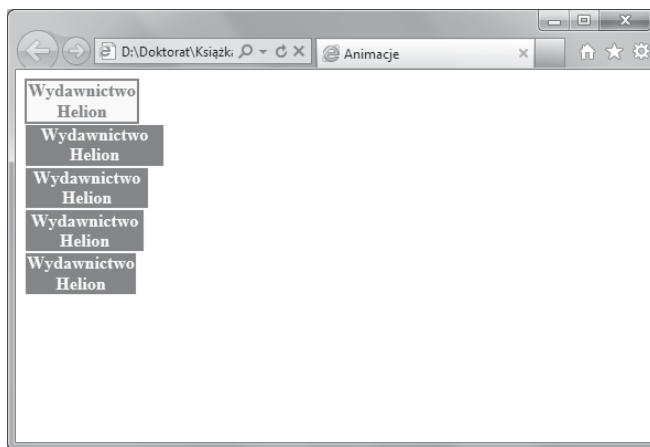
// Czy docelowa szerokość została już osiągnięta?
if (nowaSzerokosc != szerokoscDocelowa) {
    // Jeśli nie, to animuj dalej...
    element.animacjaZmianySzerokosci = setTimeout("animujSzerokosc('" + idElementu
        + "', " + rozszerzanie
        + ", " + szerokoscPoczatkowa
        + ", " + szerokoscDocelowa
        + ", " + krokZmianySzerokosci
        + ");", 10);
}
else {
    // W przeciwnym wypadku przerwij animację
    zatrzymajAnimacje(element.animacjaZmianySzerokosci);
}
}

```

JavaScript jest językiem dynamicznym, co umożliwia dynamiczną zmianę definicji obiektów w trakcie interpretacji poleceń. Wykorzystałem tę właściwość do uzupełnienia definicji animowanych obiektów o pole `animacjaZmianySzerokosci`, które służy do przechowania informacji o tym, czy dany element jest aktualnie animowany. Dzięki temu witryna działa poprawnie (rysunek 10.3).

Rysunek 10.3.

Animacja zmiany rozmiaru działająca teraz niezależnie dla każdego elementu HTML. Dzięki dynamicznemu charakterowi języka JavaScript informacja o aktualnym stanie animacji została zapisana w definicji odpowiedniego obiektu HTML



Ruch

Projekt animacji elementów HTML rozpocznę od zaimplementowania funkcji, która „przesuwa” wybrane obiekty drzewa DOM. Innymi słowy, projektowana procedura będzie modyfikowała współrzędne określające położenie tych elementów. Do tego celu wykorzystam właściwości `style.position`, `style.left` oraz `style.top`. Pierwsza z nich posłuży mi do skonfigurowania sposobu pozycjonowania wybranego obiektu HTML na względny (ang. *relative*). Dzięki temu dany element drzewa DOM będzie mógł zostać przesunięty w dowolne miejsce witryny względem swojej pozycji początkowej.

Właściwości `style.left` oraz `style.top` pozwalają zdefiniować współrzędne (w pikselach) określające odległość od lewego górnego narożnika witryny do lewego górnego narożnika wybranego obiektu HTML. Należy pamiętać, że wartości `style.left` oraz `style.top` i wszystkie pozostałe służące do konfiguracji współrzędnych wykorzystują komputerowy układ współrzędnych, którego początek znajduje się w lewym górnym rogu ekranu. Kierunek wzrostu odciętych pozostaje niezmieniony, natomiast wartości rzędnych rosną wraz z rosnącą odległością od górnej krawędzi ekranu.

Biorąc pod uwagę te ogólne zasady, definicja funkcji, której zadaniem jest zmiana położenia wybranego elementu drzewa DOM, może mieć postać przedstawioną na listingu 10.7. W celu zaprezentowania działania tej procedury powiążę ją ze zdarzeniem `onmouseover` bloku o identyfikatorze `blok6` (listing 10.8). Po umieszczeniu kursora myszy w obrębie tego bloku zostanie on przesunięty o wektor `[200 px, 200 px]`, czyli do punktu o współrzędnych:

$$x_n = x + 200 \text{ [px]}$$

$$y_n = y + 200 \text{ [px]}$$

gdzie punkt o współrzędnych (x, y) jest pozycją początkową bloku o identyfikatorze `blok6`, którą w JavaScript można odczytać za pomocą własności `clientWidth` oraz `clientHeight`.

Listing 10.7. *Zmiana położenia wybranego elementu drzewa DOM*

```
function przesunElement(idElementu, x, y) {
    var element = document.getElementById(idElementu);

    if (!element) return;

    var szerokosc = parseInt(element.clientWidth);
    var wysokosc = parseInt(element.clientHeight);

    // Czy nowe współrzędne nie spowodują wysunięcia obiektu poza dostępny obszar?
    // Jeśli tak, to skoryguj nowe współrzędne
    if (x + szerokosc > window.innerWidth)
        x = window.innerWidth - szerokosc;

    if (y + wysokosc > window.innerHeight)
        y = window.innerHeight - wysokosc;

    element.style.position = "absolute";
    element.style.left = x + "px";
    element.style.top = y + "px";
}
```

Listing 10.8. *Umieszczenie kursora myszy w obrębie obszaru zajmowanego przez obiekt `blok6` spowoduje jego przesunięcie o wektor `[200 px, 200 px]`*

```
<div id="blok6" class="zolty" onmouseover="przesunElement('blok6', 200,
↪200)">Wydawnictwo Helion</div>
```

Na podstawie przedstawionej powyżej metody przemieszczania obiektów drzewa DOM można implementować procedury, których celem jest animacja ruchu obiektów HTML. W moim przykładzie wybrany element drzewa DOM będzie poruszał się równoległe do osi odciętych. Ruch będzie wykonywany cyklicznie do momentu jego zatrzymania. Do tego celu wykorzystam metodę `przesunElement` oraz zestaw funkcji `setTimeout` i `clearTimeout`:

1. Uzupełnij skrypt witryny *Animacje.htm* o definicję funkcji `ruchPoziomy` z listingu 10.9.

Listing 10.9. *Implementacja animacji ruchu wybranego elementu drzewa DOM wzdłuż osi odciętych*

```
function ruchPoziomy(idElementu, poczatek, koniec, krok) {
    var element = document.getElementById(idElementu);

    if (!element) return;

    // Korekcja parametrów początkowych
    var szerokoscElementu = element.clientWidth;

    if (poczatek < 0) poczatek = 0;
    if (poczatek > koniec) poczatek = koniec;

    if (koniec + szerokoscElementu > window.innerWidth)
        koniec = window.innerWidth - szerokoscElementu;

    if (koniec < poczatek) koniec = poczatek;

    if (!element.aktualnaPozycja)
        element.aktualnaPozycja = element.clientLeft;

    // Wartość zmiennej nowaPozycja zależy od tego,
    // czy element powraca do pozycji początkowej,
    // czy porusza się w kierunku pozycji końcowej
    var nowaPozycja = element.powrot ? element.aktualnaPozycja - krok :
        element.aktualnaPozycja + krok;

    element.aktualnaPozycja = nowaPozycja;

    // Przesunięcie odbywa się równoległe do osi odciętych.
    // Z tego powodu współrzędna y punktu docelowego nie ulega zmianie
    przesunElement(idElementu, nowaPozycja, element.style.top);
    element.animacja = setTimeout("ruchPoziomy('" + idElementu + "', " + poczatek
        + ", " + koniec + ", " + krok + ")", 10);

    if (nowaPozycja >= koniec)
        element.powrot = true;
    else if (nowaPozycja <= poczatek)
        element.powrot = false;
}

function zatrzymajRuchPoziomy(idElementu) {
    var element = document.getElementById(idElementu);

    if (!element) return;

    zatrzymajAnimacje(element.animacja)
}
```

2. W sekcji <body> witryny *Animacje.htm* umieść definicje dwóch przycisków oraz bloku o identyfikatorze blok7 (listing 10.10).

Listing 10.10. Definicja animowanego bloku oraz przycisków, których domyślne metody zdarzeniowe uruchamiają i zatrzymują animację ruchu wzdłuż osi OX

```
<div id="blok7" class="brazowy">Poruszający się element</div>

<p>
  <input id="button1" type="button" value="Rozpocznij ruch poziomy"
  ↳onclick="ruchPoziomy('blok7', 0, window.innerWidth, 5)" />
  <input id="button2" type="button" value="Zatrzymaj ruch poziomy"
  ↳onclick="zatrzymajRuchPoziomy('blok7')" />
</p>
```

3. Otwórz witrynę *Animacje.htm* w domyślnej przeglądarce internetowej.
4. Kliknij przycisk z etykietą *Rozpocznij ruch poziomy*. Wynik działania aplikacji powinien być analogiczny do tego z rysunku 10.4.



Rysunek 10.4. Animacja ruchu wzdłuż osi OX

Naturalnym rozszerzeniem funkcjonalności zaimplementowanej w tym podrozdziale będzie zaprojektowanie funkcji, której zadaniem będzie animacja ruchu wzdłuż osi rzędnych oraz w dwóch kierunkach jednocześnie. Pozostawiam to jednak jako zadanie dla czytelnika.

Podsumowanie

Animacje są często spotykanym elementem interaktywnych aplikacji internetowych. W tym rozdziale pokazałem, w jaki sposób można je zaimplementować za pomocą technologii JavaScript. Nie ulega wątpliwości fakt, że dzięki popularnym bibliotekom JavaScript, jak chociażby jQuery, która jest zintegrowana z domyślnym szablonem aplikacji ASP.NET MVC, tworzenie i aplikowanie animacji jest znacznie prostsze niż samodzielne ich implementowanie. Jednakże w sytuacji, gdy dostępne sposoby animacji są niewystarczające na potrzeby konkretnego projektu, samodzielna umiejętność ich implementacji może okazać się nieoceniona.

Część II

Biblioteka jQuery

Skorowidz

A

adres URL, 107, 140
AJAX, 14, 107, 198
animacja, 143, 147, 159, 170
 efekty wbudowane, 263
 jQuery, 189
 kalendarza, 219
 kolorów, 258, 261
 przezroczystości, 143
 rozmiar, 147
 ruch, 151, 266
ASP.NET, 14
 Web Forms, 14
attribut
 href, 79
 id, 65, 66
 onload, 75
 src, 49
 tag, 66

B

biblioteka, 49
 jQuery, *Patrz:* jQuery
 jQuery UI, *Patrz:* jQuery UI
 tworzenie, 49
błąd, 27
 kompilacji, 16
breakpoints, *Patrz:* punkt przerwania
brudnopis, 61, 63

C

ciasteczko, cookie, 138
CSS, 54, 61, 73, 123, 124, 143, 171, 258, 269
 selektor, *Patrz:* selektor
 właściwość, 189
czujka, 57, 61

D

dane
 tabela, *Patrz:* tabela danych
 walidacja, 14, 115, 183
data, 214, 217, 225
debugowanie, 58
dokument, 90
DOM, 100, 105, 143
 Document Object Model, 100
 operacje na elementach, 176
 przeszukiwanie, 159
drag&drop, 241, 246
drzewo, *Patrz:* DOM

E

easing, 266
ECMA, 13
ECMAScript, 13
element potomny, 175
escape character, *Patrz:* znak modyfikacji

F

Firebug, 61
focus, 90
formularz, 90, 91, 117, 159, 195
 walidacja, 115, 116, 123, 183
 wysyłanie, 116, 125
 wyszukiwanie elementów, 198
funkcja, 18, *Patrz też:* metoda
 boolowska, 162
 definicja, 16, 18
 implementowane inline, 185
 parseFloat, 111
 parseInt, 111
 trygonometryczna, 127
 zwrotna, 185

G

Google Chrome, 48, 186, 187
grafika, 161

H

hiperłącze, 73, 79, 90, 226
HTML, 13, 159
kod, 54

I

instrukcja
break, 34, 41, 113
continue, 41, 42
if, 25, 26, 27
if else, 76
pętli, *Patrz:* pętla
prompt, 27
sterująca if, 24
switch, 25, 32, 41
IntelliSense, 75
interfejs
Metro, 15
użytkownika, 159, 170, 241
przenoszenie elementu, 241
upuszczanie elementu, 246
Internet Explorer, 46, 107, 108, 143
Internet Explorer 9, 53

J

JavaScript, 13
dyskretny, 159
kod źródłowy, *Patrz:* kod źródłowy
kompilacja, 16
jQuery, 14, 159, 171
animacja, *Patrz:* animacja jQuery
historia, 160
importowanie, 160
mobile, 160
wtyczka, 204, 209, 210
Accordion, 210, 238
Autocomplete, 210
Button, 210, 226
DatePicker, 210, 214, 215, 217, 218, 219, 225
Dialog, 210, 219
pager, 204
ProgressBar, 210
slider, 212
Slider, 210

tabblesorter, 204, 206
Tabs, 210, 232, 233, 238
jQuery UI
system zarządzania wersjami, 216
JScript, 13

K

kalendarz, 215, 217
animacja, 219
karta, 232
kaskadowe arkusze stylów, *Patrz:* CSS
klasa
dodawanie, 260
przełączanie, 258
usuwanie, 260
zamiana, 260
klawiatura, 94
klawisz, 96
wcisnięty, 90
kod
maszynowy, 16
pośredni, 16
źródłowy, 16
kolor, 261
komentarz, 17
kompilacja warunkowa, 13
komponent wyboru Checkbox, 90
konkatenacja, 17
konsola, 55, 61
kontrolka FileUpload, 90

L

liczba
pseudolosowa, 127
stałoprzecinkowa, 17
zmiennoprzecinkowa, 17
lista
kontekstów wykonania procedur JavaScript, 61
sortowanie, 252
literał, 21
logowanie, 14

Ł

łańcuch
tekstowy, 16, 17, 131
długość, 118
znakowy
porównywanie, 122

M

Matulewski Jacek, 212

mechanizm

drag&drop, 241

easingu, 266

menu kontekstowe, 93

metoda

\$.ajax, 186, 187

\$.get, 187

.accordion, 240

.ajax, 198

.ajaxStart, 198, 199, 201

.ajaxStop, 198, 199

.animate, 189, 190, 192, 194, 258, 261

.append, 176

.appendTo, 176

.bind, 180

.button, 226, 229

.buttonset, 226, 229

.dialog, 221, 222, 226

.draggable, 241, 242, 243, 245, 250

.droppable, 246

.effect, 266

.html, 178

.insert, 177

.isDefaultPrevented, 183

.off, 180

.on, 180

.prepend, 176

.prependTo, 176

.preventDefault, 183

.remove, 177

.resizable, 256, 257

.selectable, 250

.sortable, 253, 256

.switchClass, 260, 261

.tabs, 233, 236

.text, 178

.toggleClass, 258, 259

.trigger, 181, 182

.triggerHandler, 181, 182

.unbind, 180

.wrap, 178

addClass, 163, 260

addEventListener, 96, 99, 178

alert, 67, 136

attachEvent, 96, 99

back, 139

clearTimeout, 130

concat, 131

createElement, 100, 103

createTHead, 111

detachEvent, 99

each, 168

filter, 175

find, 169

firstChild, 100

forward, 139

GET, 107

getAttribute, 102

getDate, 130

getElementById, 66, 100

getElementByName, 66

getElementByTagName, 66

getElementsByTagName, 109

getMonth, 130

go, 139

hide, 168, 169

hover, 180, 181

HTTP, 107

insertBefore, 100

insertCell, 111

insertRow, 111

JSON.parse, 134

JSON.stringify, 137

match, 122, 131

mousedown, 184

mousemove, 184

parent, 168

parse, 134

pop, 75

POST, 107

push, 75

removeAttribute, 102, 103

removeClass, 164, 260

removeEventListener, 99

resizeTo, 87

reverse, 75

setAttribute, 102, 103

setTimeout, 130, 147

show, 169

slice, 75

sort, 75

split, 131, 132

stringify, 134

substring, 131

toggle, 169, 170

toLocaleDateString, 129

toLocaleTimeString, 129

toLowerCase, 131

toString, 136

toUpperCase, 131

window.focus, 81

window.open, 84

window.open, 81

zdarzeniowa, 172

formularza, 115

motyw, 269
 domyślny, 269
 Mozilla Firefox, 47, 53, 61, 63
 MSDN, 127
 MVC, 14
 mysz
 przycisk, 90, 91
 prawy blokowanie, 14
 wskaźnik, 90

N

NaN, 17, 111
 narzędzie deweloperskie, 53
 node, *Patrz:* węzeł
 Not a Number, *Patrz:* NaN

O

obiekt
 Array, 110, 127
 Date, 127, 129
 definicja, 16
 document, 66, 100, 161
 document.forms, 117
 event, 184
 filtrowanie, 171
 history, 100, 138, 139
 HTML, 69, 90
 JSON, 127, 134
 konstruktor, 135
 location, 100, 138, 140
 Math, 127
 navigator, 138
 potomny, 100
 String, 127, 131
 table, 73
 tworzenie, 135
 wbudowany, 127
 window, 87, 100
 XMLHttpRequest, 107, 108
 obrazu ładowanie, 90
 okno
 dialogowe, 219, 221
 modalne, 125, 219
 operacja matematyczna, 127
 operator
 &, 32
 ?., 76
 ^, 32
 |, 32
 ++, 34
 alternatywy, *Patrz:* operator |

arytmetyczny, 16
 bitowy, 32
 dekrementacji, *Patrz:* operator --
 dodawania, 17
 inkrementacji, *Patrz:* operator ++
 koniunkcji, *Patrz:* operator &
 logiczny, 24
 !, 24
 !=, 24
 &&, 25, 30, 31
 ||, 25, 29
 <=, 24
 ==, 24
 >, 24
 >=, 24, 31
 alternatywy, *Patrz:* operator logiczny ||
 koniunkcji, *Patrz:* operator logiczny &&
 priorytet, 31
 odejmowania, 17
 przedrostkowy, 34
 przyrostkowy, 34
 różnicy symetrycznej, *Patrz:* operator ^

P

pamięć, 16
 pasek postępu, 210
 pętla, 35
 do while, 37, 41, 42
 for, 38, 41
 for in, 105, 113
 while, 35, 41, 42
 PHP, 14
 plik
 .js, 49
 multimedialny, 161
 XML, 106, 108
 plug-in, *Patrz:* jQuery wtyczka
 pole
 tekstowe, 66, 90, 91
 maskowane, 90
 wyboru, 226
 polecenie, *Patrz:* instrukcja
 pozycjonowanie
 względne, 151
 procedura, 18
 protokół
 file, 108
 HTTP, 107
 transmisji, 134
 przeglądarka
 Google Chrome, *Patrz:* Google Chrome
 Internet Explorer, *Patrz:* Internet Explorer

internetowa

okno, 79, 83, 84, *Patrz:* okno

właściwości, 138

konfiguracja, 45

Mozilla Firefox, *Patrz:* Mozilla Firefox

przesłanianie lokalne, 23

przezroczystość, 145

przycisk, 36, 37, 38, 41, 42, 75, 87, 168, 173, 226

radio, 90, 226, 229

punkt

kontrolny, *Patrz:* punkt przzerwania

przerwania, 58, 59

R

ramka, 90

Resig John, 160

S

selektor, 160, 162, 165, 173, 198

serializacja SOAP, 134

Single Object Access Protocol, *Patrz:* SOAP

skrypt, 15

czas realizacji, 54

debugowanie, *Patrz:* debugowanie

implementowanie, 53

język, 15

kliencki, 13, 14, 15, 45

miejsce wykonania, 15

serwerowy, 14, 15

śledzenie, 58

typ, 15

uruchamiany

po stronie klienta, *Patrz:* skrypt:kliencki

po stronie serwera, *Patrz:* skrypt:serwerowy

w konsoli, 56

słownik, 137

słowo kluczowe

case, 33

function, 18

new, 110

return, 18

switch, 24, 25, *Patrz też:* instrukcja switch

this, 72

var, 16, 17, 21, 135

SOAP, 134

stała matematyczna, 127

stos wywołań, 61

strona internetowa, 49

suwak, 212

T

tabela danych, 110

nagłówek, 111

sortowanie, 204

tablica asocjacyjna, 137

tekstu wyrównywanie, 75

ThemeRoller, 269

U

unobtrusive JavaScript, *Patrz:* JavaScript
dyskretny

W

warstwa

funkcjonalna, 159

prezentacji, 159

węzeł, 100

atrybuty, 102

tworzenie, 102

witryny modyfikacja dynamiczna, 13

właściwość

activeClass, 248

backgroundColor, 261

borderBottomColor, 261

borderLeftColor, 261

borderRightColor, 261

borderTopColor, 261

button, 184

checked, 83

childNodes, 109

className, 124

clientHeight, 152

clientWidth, 152

color, 261

disabled, 84

firstChild, 109

hash, 140

height, 191

host, 140

hostname, 140

hoverClass, 248

href, 140

innerHeight, 87

innerWidth, 87

lastChild, 109

left, 191

length, 139

name, 119

właściwość

- nodeValue, 109
- opacity, 143, 145
- outlineColor, 261
- pathname, 140
- port, 140
- protocol, 140
- responseText, 107
- responseXML, 107, 108
- revert, 250
- screen.availHeight, 87
- screen.availWidth, 87
- search, 140
- selectedIndex, 119
- style, 143
- style.display, 77
- style.filter, 143
- style.left, 151
- style.opacity, 143
- style.position, 151
- style.top, 151
- top, 191
- value, 119
- width, 191

wyrażenie regularne, 120, 162

Z

zakładka, *Patrz*: karta

zdarzenie, 178, 213

- \$(document).ready, 186

- argumenty, 184

- change, 198, 214

- click, 168

- dotyczące

- klawiatury, 89

- myszy, 89

- witryny, 89

- drag, 243

- hover, 259

- mouseout, 147

- obsługa

- aktywacja, 180

- deaktywacja, 180

- domyślna, 183

- onabort, 90

- onblur, 90

- onchange, 90, 122

- onclick, 90

- oncontextmenu, 93

- ondblclick, 90

- ondragdrop, 90

- onerror, 90

- onfocus, 90

- onkeydown, 89, 90, 94

- onkeypress, 90

- onkeyup, 89, 90

- onload, 89, 91, 97

- onmousedown, 89, 90, 93

- onmouseout, 89, 90

- onmouseover, 89, 90, 152

- onmouseup, 90

- onmove, 90

- onreset, 90, 125

- onresize, 90

- onselect, 91

- onSelect, 219, 225

- onsubmit, 91, 116, 125

- onunload, 89, 91

- ready, 161

- rejestracja, 179

- slide, 214

- start, 243

- stop, 243

- window.onload, 161

- wyzwolenie, 181

zmienna, 16

- deklaracja, 17

- globalna, 17, 21

- przykrywanie, *Patrz*: przesłanianie lokalne

- lokalna, 17, 61, 135

- deklaracja, 16

- śledzenie, 57, 60

- tablicowa, 75

- deklaracja, 75

- typ

- deklaracja, 16

- liczbowy, 16

- tekstowy, 16

- znakowy, 16

znacznik

- <a>, 79

-
, 21

- <script>, 15, 16, 45, 49

- <table>, 100

- <td>, 100

- <tr>, 100

- <u>, 21

- HTML, 100, 210

- przejścia do nowej linii,

- Patrz*: znacznik

- XML, 108

znak

- [], 75

- {}, 18

- backslash, *Patrz*: znak ukośnik wsteczny

- cudzysłów, 21

- modyfikacji, 20

\n, 20
nawias kwadratowy, *Patrz*: znak []
podwójny ukośnik, 17
slash, *Patrz*: znak ukośnik
specjalny, 21
\t, 20
ukośnik wsteczny, 21

Ż

zadanie
asynchroniczne, 159
GET, 187
HTTP, 107, 187

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



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

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

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

GRUPA WYDAWNICZA

 **Helion SA**

JavaScript i jQuery

Kompletny przewodnik dla programistów
interaktywnych aplikacji internetowych w Visual Studio

Technologia JavaScript nie od dziś jest uważana za potężne narzędzie, doskonale wpasowujące się w sposób działania aplikacji internetowych po stronie serwera. To zaś jest zaleta nie do przecenienia w czasach, gdy przez globalną sieć przesyłane są niewiarygodnie wielkie ilości danych, a internet w zasadzie służy do załatwiania wszelkich codziennych spraw. Każda współczesna aplikacja internetowa musi zapewniać szybki transfer i bezpieczeństwo danych, a także oferować użytkownikom końcowym wygodny interaktywny widok wraz z możliwością otwierania i wprowadzania danych na różnych urządzeniach. Połączenie JavaScriptu i jQuery pozwala osiągnąć taki stan rzeczy bez nadmiernego obciążania sieci i serwerów.

W tej książce znajdziesz informacje o technologii JavaScript oraz bibliotekach jQuery oraz jQuery UI, które umożliwiają tworzenie interaktywnych widoków aplikacji internetowych. Dowiesz się więcej o składni języka JavaScript i zapewnianiu interakcji z różnymi elementami strony oraz o tworzeniu i wykorzystywaniu bibliotek skryptów. Nauczysz się zmieniać właściwości różnych elementów witryny, odkryjesz, do czego służą formularze i obiekty wbudowane, a ponadto zrozumiesz, jak ważną rolę w procesie projektowania i implementowania aplikacji internetowych odgrywają biblioteki jQuery i jQuery UI. Warto, przekonaj się sam!

- Konfiguracja obsługi JavaScriptu i bibliotek skryptów
- Debugowanie kodu w JavaScriptcie
- Dostęp do wybranych elementów stron internetowych
- Obsługa zdarzeń i drzewo DOM
- Tabele danych, źródła XML oraz pętla for in
- Formularze, obiekty wbudowane i animacje
- Podstawy jQuery: selektory, zdarzenia, programowanie asynchroniczne
- Zaawansowane aspekty jQuery: animacje, formularze, AJAX, wtyczki i rozszerzenia
- jQuery UI: wtyczki, interakcja z komponentami oraz zaawansowane efekty animacji

JSIQKP_okładka.indd 1

Interaktywna strona WWW?

Tylko z JavaScriptem i jQuery!

helion.pl
księgarnia
internetowa

Nr katalogowy: 15499

Księgarnia internetowa:

<http://helion.pl>

Zamówienia telefoniczne:

0 801 339900

0 601 339900



Helion

Sprawdź najnowsze promocje:

- <http://helion.pl/promocje>
- Książki najchętniej czytane:
- <http://helion.pl/bestsellery>
- Zamów informacje o nowościach:
- <http://helion.pl/novosci>

Helion SA

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

e-mail: helion@helion.pl

<http://helion.pl>

sięgnij po WIĘCEJ



KOD KORZYŚCI

ISBN 978-83-246-8283-6



9 788324 682836

Cena: 49,00 zł

Informatyka w najlepszym wydaniu