

Najlepszy podręcznik do jQuery!

# Rusz głową!

# jQuery



Szybko implementuj złożone formularze HTML



Dodawaj interaktywność do swoich stron w zaledwie kilku wierszach kodu

Buduj aplikacje internetowe bez stosowania irytujących wtyczek



Twórz własne, niestandardowe funkcje jQuery



Korzystaj z animacji i Ajaksa na swoich stronach

O'REILLY®

Ryan Benedetti, Ronan Cranley



Tytuł oryginału: Head First jQuery

Tłumaczenie: Ireneusz Jakóbk

ISBN: 978-83-246-3864-2

© 2012 Helion S.A.

Authorized Polish translation of the English edition of *Head First jQuery 9781449393212*  
© 2011 Ryan Benedetti and Ronan Cranley.

This translation is published and sold by permission of O'Reilly Media, Inc., which owns or controls all rights to publish and sell the same.

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

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

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

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

Wydawnictwo HELION  
ul. Kościuszki 1c, 44-100 GLIWICE  
tel. 32 231 22 19, 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
WWW: <http://helion.pl> (księgarnia internetowa, katalog książek)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

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

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

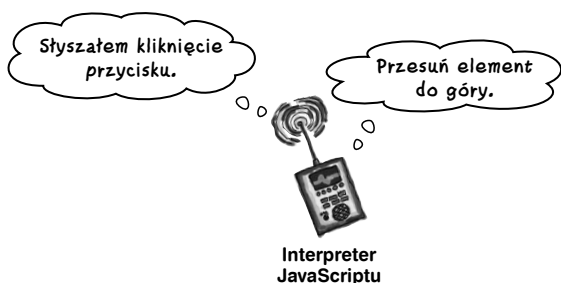
# 1

## Zaczynamy z jQuery

### Akcja na stronach internetowych

**Chcesz więcej dla swoich stron.** Masz w zanadrzu HTML oraz CSS i chciałbyś wzbogacić swoje umiejętności o pisanie skryptów, ale nie masz zamiaru spędzić całego życia, pisząc je wiersz po wierszu. Potrzebujesz biblioteki skryptów, która pozwoli Ci zmieniać strony internetowe w locie. A skoro wyrażamy już nasze życzenia, to może dodać jeszcze współpracę z Ajaxem i PHP? Czy da się zrobić w trzech wierszach to, do czego większość języków po stronie klienta potrzebuje piętnastu? Pobożne życzenia! W żadnym wypadku! Musisz zapoznać się z jQuery.

	<p>Chcesz włączyć strony 38</p> <p>HTML i CSS są w porządku, ale... 39</p> <p>...potrzebujesz mocy skryptu 40</p> <p>Nadchodzi jQuery (i JavaScript)! 41</p> <p>Spojrzenie do wnętrza przeglądarki 43</p> <p>Ukryta struktura strony 44</p> <p>Dzięki jQuery drzewo DOM nie jest takie straszne 45</p> <p>Jak to działa? 47</p> <p>jQuery wybiera elementy tak samo jak CSS 49</p> <p>Stylu, przedstawiam ci skrypt 50</p> <p>Selektory jQuery do usług 51</p> <p>jQuery przetłumaczony 52</p> <p>Twoja pierwsza fucha w jQuery 56</p> <p>Skonfiguruj swoje pliki HTML i CSS 60</p> <p>Wysuń się... 62</p> <p>Niechaj bładość będzie z Tobą 63</p> <p>Uratowałeś kampanię Kudłaci Przyjaciele 66</p> <p>Twój niezbędnik jQuery 69</p>
--	---

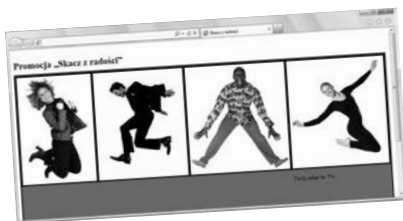


## 2

## Selektory i metody

## Chwytaj i w drogę

**jQuery pomaga chwytać elementy strony internetowej i robić z nimi różne rzeczy.** W tym rozdziale zagłębimy się w selektory i metody jQuery. Dzięki selektorom możemy chwytać elementy na stronie, a metody pozwalają na robienie z nimi rozmaitych sztuczek. Podobnie jak wielka księga zaklęć magicznych, biblioteka jQuery pozwala nam zmieniać masę rzeczy w locie. Możemy sprawiać, że obrazy rozplywają się w powietrzu i pojawiają znikąd. Możemy wybrać określony fragment tekstu i w animowany sposób zmienić rozmiar jego czcionki. Rozpoczniemy więc pokaz — złapmy kilka elementów strony i do roboty!



„Skacz z radości” potrzebuje Twojej pomocy	72
Jakie są wymogi projektu?	73
Zagłębiamy się w sekcje	75
Zdarzenie click w zbliżeniu	78
Dodaj do strony metodę click	81
Bądź bardziej konkretny	83
Przydzielanie elementów do klas	84
ID-entyfikowanie elementów	85
Pospinaj swoją stronę	88
Tymczasem na naszej liście	91
Tworzenie miejsca w pamięci	92
Łącz za pomocą konkatenacji	93
Tymczasem w naszym kodzie...	94
Wstaw komunikat za pomocą metody append	95
Wszystko działa świetnie, ale...	97
Daj mi \$(this)	99
Zapręgnij \$(this) do pracy	100
Pozbywaj się na dobre za pomocą remove	102
Schodzimy w dół z selektorem potomka	103
Twoja kolej na skakanie z radości	109
Twój niezbędnik jQuery	110

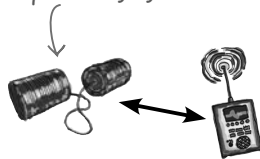
# 3

## Zdarzenia i funkcje jQuery

### Niech się coś dzieje na Twojej stronie

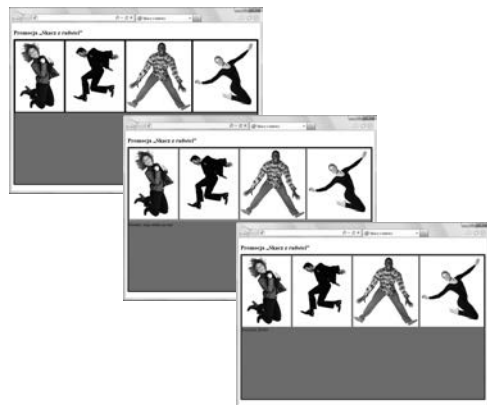
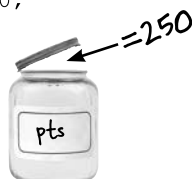
**jQuery ułatwia dodawanie akcji oraz interaktywności do każdej strony internetowej.** W tym rozdziale przyjrzymy się temu, jak sprawić, żeby strona reagowała, kiedy użytkownicy wchodzi z nią w interakcję. Sprawienie, że kod będzie reagował na działania użytkownika, wyniesie Twoją witrynę na zupełnie nowy poziom. Zajmiemy się też tworzeniem funkcji wielokrotnego użycia, dzięki czemu kod napiszesz raz, ale już korzystać z niego będziesz wielokrotnie.

Stuchacz zdarzeń  
słyszy zdarzenie  
i przekazuje je do...



...interpretera  
JavaScriptu, który  
sprawdza, co trzeba  
zrobić po każdym  
zdarzeniu...

```
var pts = 250;
```



Twoja znajomość jQuery znowu jest potrzebna	112
Facet od pieniędzy ma rację...	113
Dodawanie zdarzeń na stronę	115
Za kulisami słuchacza zdarzeń	116
Wiązanie zdarzenia	117
Wyzwalanie zdarzenia	118
Usuwanie zdarzenia	122
Weźmy się za <del>bar</del> elementy	126
Struktura Twojego projektu	132
Dodajemy funkcje	136
Śrubki i trybiki funkcji	137
Funkcja anonimowa	138
Funkcje nazwane jako funkcje obsługi zdarzeń	139
Przekazywanie zmiennej do funkcji	142
Funkcja może też zwracać wartość	143
Korzystaj z logiki warunkowej w celu podejmowania decyzji	145
„Skacz z radości” potrzebuje jeszcze większej pomocy	149
Metody mogą zmieniać CSS	151
Dodaj zdarzenie hover	153
To już prawie wszystko...	155
Twój niezbędnik jQuery	158

## 4

## Manipulowanie stroną internetową za pomocą jQuery

## Zmodyfikuj drzewo DOM

**To, że strona skończyła się pobierać, nie znaczy od razu, że musi ona zachowywać tę samą strukturę.** W rozdziale 1. zobaczyliśmy, jak podczas wczytywania strony, w celu ustalenia jej struktury, jest konstruowane drzewo DOM. W bieżącym rozdziale dowiemy się, jak poruszać się w górę i w dół struktury drzewa DOM i jak pracować z hierarchią elementów oraz relacjami rodzic – dziecko, aby za pomocą jQuery zmieniać w locie strukturę strony.



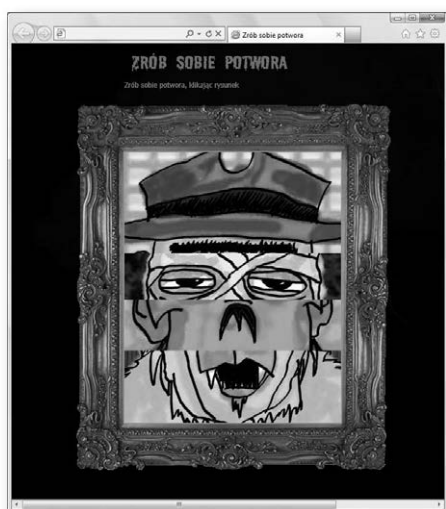
Restauracja w Webowicach chce interaktywnego menu	160
Dla vegetarian	161
Poklasyfikuj swoje elementy	166
Pora na przyciski	169
Co teraz?	171
Biegamy po gałęziach drzewa DOM	176
Wspinaczkowe metody przechodzenia drzewa	177
Metody łańcuchowe — aby sięgać jeszcze dalej	178
Zmienne mogą przechowywać też elementy	185
I znowu ten dolar...	186
Poszerz swoje możliwości przechowywania o tablice	187
Przechowuj elementy w tablicy	188
Wymieniaj elementy za pomocą replaceWith	190
Jak może nam pomóc replaceWith?	191
Zastanów się, zanim użyjesz replaceWith	193
replaceWith nie sprawdza się w każdej sytuacji	194
Wstawienie zawartości HTML do drzewa DOM	195
Użyj metod filtrujących w celu zawężenia wyboru (część 1.)	197
Użyj metod filtrujących w celu zawężenia wyboru (część 2.)	198
Przywróć hamburgera	201
A co z mięsem?	202
Treściwa tablica	203
Metoda each przegląda tablice	204
To już wszystko... prawda?	207
Twój niezbędnik jQuery	210

# 5

## Efekty i animacja w jQuery

### Nieco wdzięku na Twojej stronie

**Sprawianie, że na Twojej stronie będą się dziać różne rzeczy, to niezły pomysł, ale tylko do czasu.** Jeśli nie będzie ona wyglądać atrakcyjnie, ludzie nie zechcą z niej korzystać. Dlatego właśnie przydadzą Ci się efekty i animacja w jQuery. W bieżącym rozdziale nauczysz się, co zrobić, aby wraz z upływem czasu elementy zmieniały swoje miejsce na stronie, jak pokazać lub ukryć określony element albo jak go powiększyć lub zmniejszyć, a wszystko to na oczach użytkownika. Dowiesz się też, jak zakładać harmonogramy animacji, aby uruchamiały się one w różnych odstępach czasu, co nada Twojej stronie bardzo dynamiczny wygląd.



Gryzmołki potrzebują aplikacji sieciowej	212
Zrób sobie potwora	213
Zrób sobie potwora — potrzebujemy układu i położenia na stronie	214
Trochę więcej struktury i stylu	217
Uaktywniamy interfejs	218
Robimy efekt błyskawicy	223
W jaki sposób jQuery animuje elementy?	224
Efekty wygaszania zmieniają właściwość CSS opacity	225
Wysuwanie sprowadza się do zmiany wysokości	226
Zatrudnij efekty wygaszania	228
Łącz efekty za pomocą łańcuchów metod	229
Kontratakujemy za pomocą funkcji czasowych	230
Dodaj do swojego skryptu funkcje błyskawicy	233
Efekty własnej roboty wykorzystujące metodę animate	235
Co można, a czego nie można animować?	236
Metoda animate zmienia w czasie styl	238
Dokładnie skąd dokąd?	241
Ruch bezwzględny obiektów a ich ruch względny	242
Przesuwaj elementy względnie dzięki łączeniu operatorów	243
Dodaj funkcje animujące do swojego skryptu	245
Patrz, mam! Bez Flasha!	248
Twój niezbędny jQuery	250



## 6

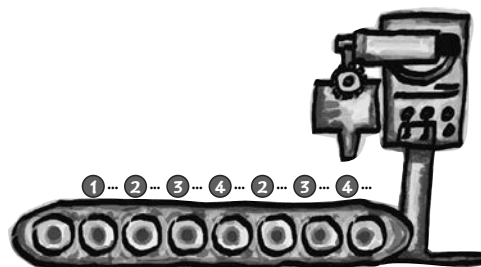
## jQuery i JavaScript

**-Luke jQuery, jestem twoim ojcem!**

**Sama biblioteka jQuery nie da sobie ze wszystkim rady.** Chociaż jest ona biblioteką JavaScriptu, nie potrafi niestety robić wszystkiego, co jej język ojczysty. W tym rozdziale przyjrzymy się niektórym aspektom JavaScriptu, których będziesz potrzebował do tworzenia naprawdę atrakcyjnych stron, oraz temu, jak można z nich korzystać w jQuery w celu tworzenia niestandardowych list i obiektów, a także jak przechodzić przez nie w pętlach, aby ułatwić sobie życie.



Atrakcyjniejszy barek Ruszczyca Głową	252
Obiekty oferują inteligentne przechowywanie	254
Buduj własne obiekty	255
Twórz obiekty wielokrotnego użycia za pomocą konstruktorów	256
Interakcja z obiektami	257
Konfiguracja strony	258
Powrót tablic	261
Dostęp do tablic	262
Dodawaj i aktualizuj elementy tablicy	263
Zrób to jeszcze raz (i jeszcze raz, i jeszcze raz, i...)	265
Szukanie igły w stogu siana	268
Czas na podjęcie decyzji... znowu!	275
Operatory porównania i operatory logiczne	276
Czyszczenie w jQuery...	282
Dodajemy więcej emocji	286
Twój niezbędnik jQuery i JavaScriptu	288





# 7

## Niestandardowe funkcje dla niestandardowych efektów

### Co ostatnio dla mnie zrobiłeś?

**Kiedy połączysz niestandardowe efekty jQuery** z funkcjami JavaScriptu, możesz sprawić, że Twój kod — a tym samym Twoja aplikacja internetowa — będzie wydajniejszy, skuteczniejszy i *lepszy*. W tym rozdziale zagłębisz się jeszcze bardziej w ulepszanie efektów jQuery poprzez obsługę **zdarzeń przeglądarki**, pracę z **funkcjami czasu** oraz poprawę **organizacji i możliwości wielokrotnego użycia** Twoich niestandardowych funkcji w JavaScriptcie.



Nadciąga burza	290
Utworzyliśmy funkcję potwora	291
Sprawuj kontrolę nad efektami czasowymi za pomocą obiektu window	292
Odpowiadaj na zdarzenia przeglądarki za pomocą metod onblur i onfocus	295
Metody czasowe mówią funkcjom, kiedy działać	299
Napisz funkcje stopLightning i goLightning	302
Prośba o nową funkcjonalność w Zrób sobie potwora	310
Zacznijmy losować	311
Znasz już bieżącą pozycję...	312
...oraz funkcję getRandom	312
Przesuwaj <u>względnie</u> do bieżącej pozycji	316
Zrób sobie potwora v2 to prawdziwy przebój!	325
Twój niezbędnik jQuery	326



W rzeczy samej, zęby to najlepsze, co mam, ale na miłość boską — mógłbym chcieć skorzystać też z reszty mojej twarzy!

setTimeout()



setInterval()



delay()



## 8

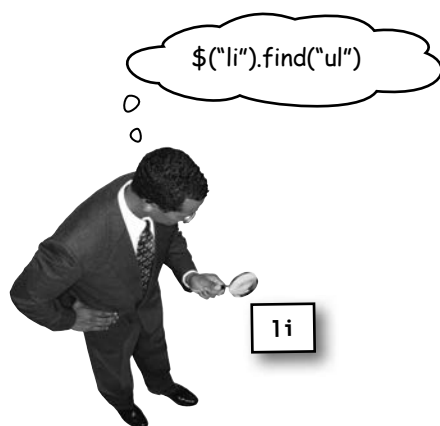
## jQuery i Ajax

## Proszę podać dane

**Używanie jQuery w celu robienia sprytnych sztuczek z CSS i drzewem DOM to fajna sprawa**, ale już wkrótce będziesz musiał odczytywać informacje (albo dane) z serwera i je wyświetlać. Może nawet przyjdzie Ci zaktualizować fragmenty strony informacjami z serwera bez potrzeby jej odświeżania. Poznaj Ajax. W połączeniu z jQuery i JavaScriptem może on właśnie coś takiego robić. W tym rozdziale dowiemy się, jak jQuery radzi sobie z namawianiem Ajaksa do wysyłania zapytań do serwera i co potem robi z otrzymanymi informacjami.



Wprowadź bieg Od bitu do bajtu w XXI wiek	328
Spojrzenie na stronę z zeszłego roku	329
Dynamizujemy	332
Stary Webie, poznaj nowego Weba	333
Zrozumieć Ajax	334
Czym jest Ajax?	334
X	335
Pobieranie danych przy użyciu metody ajax	340
Przetwarzanie danych XML	342
Harmonogramy zdarzeń na stronie	346
Funkcje samowywołujące się	347
Dostać więcej od swojego serwera	350
Która godzina?	351
Wyłączanie zaplanowanych zdarzeń na stronie	356
Twój niezbędnik jQuery i Ajaksa	360

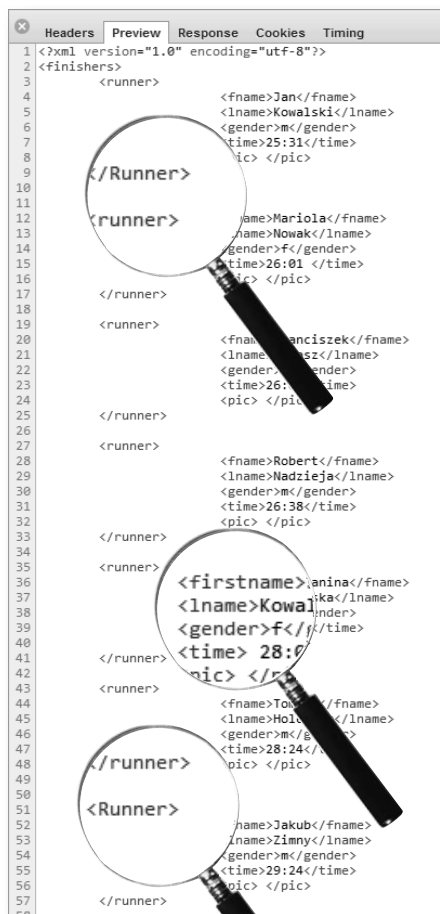


# 9

## Obsługa danych JSON

### Kliencie, oto serwer

**Chociaż czytanie danych z pliku XML jest bardzo przydatne, to jednak nie zawsze wystarcza.** Bardziej interaktywny format wymiany danych (JavaScript Object Notation, znany też jako JSON) ułatwia pobieranie danych z serwera. JSON jest też łatwiejszy do wygenerowania i czytania niż format XML. Używając jQuery, PHP oraz SQL-a, nauczysz się, jak utworzyć bazę danych przechowującą informacje, które będziesz mógł później odczytać za pomocą formatu JSON i wyświetlić na ekranie przy użyciu jQuery. Tak objawia się prawdziwa moc aplikacji internetowych!



Dział Marketingu webowickiej Megakorporacji nie zna XML-a	362
Błędy w XML-u psują stronę	363
Pobierz dane ze strony	364
Co zrobić z danymi	367
Sformatuj dane, zanim je wyślesz	368
Prześlij dane do serwera	369
Przechowuj dane w bazie MySQL	371
Utwórz bazę danych w celu przechowywania informacji o biegaczach	372
Anatomia instrukcji insert	374
Używaj PHP w celu odczytywania danych	377
Przetwórz dane POST na serwerze	378
Połącz się z bazą danych za pomocą PHP	379
Użyj polecenia select, aby pobrać dane z bazy	381
Pobierz dane za pomocą PHP	383
Z pomocą nadchodzi JSON!	386
jQuery + JSON = coś niesamowitego	387
Kilka reguł PHP..	388
Kilka (kolejnych) reguł PHP..	389
Formatowanie informacji wyjściowych w PHP	390
Dostęp do danych w obiekcie JSON	397
Sanityzacja i walidacja danych w PHP	400
Twój niezbędny jQuery, Ajaksa, PHP i MySQL	405



## 10

## Interfejs użytkownika w jQuery

## Diametralna zmiana wyglądu

**Sieć WWW żyje i umiera dzięki swoim użytkownikom i ich danym.** Zbieranie danych od użytkowników to poważne zajęcie, które może być czasochłonnym wyzwaniem dla programisty. Widziałeś już, jak Ajax, PHP i MySQL mogą pomóc w czynieniu efektywniejszym działania aplikacji internetowych. Teraz przyjrzymy się, jak jQuery może nam pomóc w utworzeniu interfejsów użytkownika dla formularzy zbierających dane. Po drodze otrzymasz sporą dawkę jQuery UI — oficjalnej biblioteki interfejsu użytkownika języka jQuery.



Kryptozoolodzy.org potrzebują zmiany wyglądu	408
Odpicuj swój formularz HTML	409
Oszczędź sobie bólu głowy (i zyskaj trochę czasu) za pomocą jQuery UI	412
Co znajduje się w środku pakietu jQuery UI	416
Utwórz kalendarz dla formularza z obserwacjami	417
jQuery za kulisami	418
Opcje widgetów są konfigurowalne	419
Nadaj styl przyciskom	422
Kontroluj pola liczbowe za pomocą suwaków	426
Komputery mieszają kolory, używając barwy czerwonej, zielonej i niebieskiej	435
Napisz funkcję refreshSwatch	438
I jeszcze pewien drobiazg...	442
Twój niezbędnik jQuery	446

Kiedy wreszcie dadzą mi spokój ci paparazzi?



# 11

## jQuery i API

### Obiekty, wszędzie obiekty

**Niezależnie od tego, jak bardzo utalentowanym programistą jesteś, sam nie dasz sobie rady ze wszystkim...** Zobaczyliśmy, jak można dołączać wtyczki jQuery, takie jak jQuery UI, albo nawigację z wykorzystaniem kart, aby bez większego wysiłku wzbogacać programy pisane w jQuery. W celu wyniesienia naszych aplikacji na jeszcze wyższy poziom, dodania do nich paru z naprawdę niezłych narzędzi dostępnych w Internecie oraz skorzystania z informacji udostępnionych przez prawdziwych potentatów — takich jak Google, Twitter albo Yahoo! — będziemy jednak potrzebować czegoś... więcej. Firmy te, i wiele innych, dostarczają API (*application programming interfaces*, czyli interfejsy programowania aplikacji) dedykowane dla swoich usług, z których możesz korzystać w swojej witrynie. W tym rozdziale przyjrzymy się podstawom API i zastosujemy niezwykle popularny interfejs, jakim jest Google Maps API.

Cześć! Jestem markerem. Chętnie zamarkuję... to jest zawrę z Tobą znajomość. Na pewno już gdzieś mnie widziałeś!



Gdzie jest <del>generał</del> paskuda?	448
API Google Maps	450
API używają obiektów	451
Dołącz mapy Google do swojej strony	453
Odczytywanie danych w formacie JSON za pomocą SQL-a i PHP	456
Punkty na mapie to markery	460
Lista kontrolna funkcji wyświetlającej wiele obserwacji	464
Nasłuchiwanie zdarzeń na mapie	474
Udało się!	478
Twój niezbędnik jQuery API	481



## A

## Ostatki

**Dziesięć najważniejszych rzeczy  
(o których nie napisaliśmy)**

**Nawet po tym wszystkim ciągle jest jeszcze wiele rzeczy, do których nie dotarliśmy.** Istnieje mnóstwo dobrodziejstw jQuery i JavaScriptu, których nie udało nam się wcisnąć do tej książki. Byłoby nieuczciwością nie wspomnieć o nich, a tak będziesz lepiej przygotowany na każdy inny aspekt jQuery, który możesz napotkać podczas swoich podróży.

1. Wszystkie drobiazgi z biblioteki jQuery	484
2. jQuery w sieciach CDN	487
3. Przestrzeń nazw jQuery: metoda noConflict	488
4. Debugowanie kodu jQuery	489
5. Zaawansowane techniki animacji: kolejki	490
6. Walidacja formularzy	491
7. Biblioteka jQuery UI Effects	492
8. Tworzenie własnych wtyczek do jQuery	493
9. JavaScript dla zaawansowanych: domknięcia	494
10. Szablony	495

# B

## Konfiguracja środowiska programistycznego

### Przygotuj się do przyszłych sukcesów

**Potrzebujesz miejsca na ćwiczenie nowo nabytych umiejętności PHP bez narażania swoich danych na niebezpieczeństwa czyhające w sieci.** Posiadanie bezpiecznego miejsca na opracowywanie aplikacji PHP przed wypuszczeniem jej w szeroki świat sieci internetowej zawsze jest dobrym pomysłem. Dodatek ten zawiera instrukcje dotyczące instalacji serwera WWW, MySQL i PHP, dzięki czemu będziesz mieć do dyspozycji bezpieczne miejsce do pracy i ćwiczeń.

Utwórz środowisko programistyczne PHP	498
Dowiedz się, co już masz	498
Czy masz serwer WWW?	499
Czy masz PHP? Którą wersję?	499
Czy masz MySQL? Którą wersję?	500
Zacznij od serwera WWW	501
Zakończenie instalacji serwera Apache	502
Instalacja PHP	502
Etapy instalacji PHP	503
Etapy instalacji PHP: zakończenie	504
Instalacja MySQL	504
Etapy instalacji MySQL w systemie Windows	505
Uaktywnienie PHP w systemie Mac OS X	510
Etapy instalacji MySQL w systemie Mac OS X	510

# S

## Skorowidz

518





## 5. Efekty i animacja w jQuery

# Nieco wdzięku na Twojej stronie

Zobaczcie, jak pięknie się poruszam. Mam w sobie tyle gracji. Założę się, że tak nie potraficie!



**Sprawianie, że na Twojej stronie będą się dziać różne rzeczy, to niezły pomysł, ale tylko do czasu.** Jeśli nie będzie ona wyglądać atrakcyjnie, ludzie nie zechcą z niej korzystać. Dlatego właśnie przydadzą Ci się efekty i animacja w jQuery. W bieżącym rozdziale nauczysz się, co zrobić, aby wraz z upływem czasu elementy zmieniały swoje miejsce na stronie, jak pokazać lub ukryć określony element albo jak go powiększyć lub zmniejszyć, a wszystko to na oczach użytkownika. Dowiesz się też, jak zakładać harmonogramy animacji, aby uruchamiały się one w różnych odstępach czasu, co nada Twojej stronie bardzo dynamiczny wygląd.

## Gryzmołki potrzebują aplikacji sieciowej

Gryzmołki zaopatrują webowickie dzieci w atrakcyjne materiały artystyczne do zabawy. Kilka lat temu firma uruchomiła popularną witrynę internetową, w której udostępniła interaktywne aplikacje dla dzieci. Społeczność jej fanów rośnie w takim tempie, że Gryzmołki zaczęły mieć problemy z radzeniem sobie na bieżąco z napływającymi prośbami.

Aby zaspokoić potrzeby nowej, większej klienteli Gryzmołków, dyrektor ds. projektów internetowych chciałaby utworzyć aplikację, która nie korzystałaby z Flasha ani z żadnych innych wtyczek przeglądarkowych.

**GRYZMOŁKI**



Nasze projekty dla dzieciaków to przede wszystkim dobra zabawa i zaangażowanie. Czy możesz wykonać dla nas aplikację dla grupy wiekowej od 6 do 10 lat? Potrzebujemy mnóstwa efektów wizualnych i trochę interaktywności. Tylko bez Flasha proszę!

## Zrób sobie potwora

Oto plan aplikacji przekazany przez dyrektora ds. projektów internetowych wraz z plikami graficznymi od projektanta przeznaczonymi do użycia w programie:

### Projekt Zrób sobie potwora

Zadaniem aplikacji Zrób sobie potwora jest zabawianie dzieci w docelowej grupie wiekowej przez umożliwienie im zrobienia sobie potwora dzięki mieszaniu 10 różnych głów, oczu, nosów i ust. Przejścia między poszczególnymi częściami twarzy potwora powinny być animowane.

#### Interfejs użytkownika

**Pojemnik**

**Ramka**

**Obszar głowy**  
*Kliknij, aby przewinąć głowę potwora.*

**Obszar oczu**  
*Kliknij, aby przewinąć oczy potwora.*

**Obszar nosa**  
*Kliknij, aby przewinąć nos potwora.*


**Obszar ust**  
*Kliknij, aby przewinąć usta potwora.*

**img - lightning1**  
**img - lightning2**  
**img - lightning3**


*Po dziewięciu kliknięciach każdy pasek powinien „przewinąć się” na początek.*

#### Animacja

Symulacja obrazująca, jak powinna się zmieniać część twarzy potwora.



Symulacja obrazująca, jak powinna wyglądać animacja błyskawicy.




*Obrazy błyskawicy powinny pojawiać się i znikać szybko, tak jakby rzeczywiście się błyskało.*


---

#### Pliki graficzne


*frame.png*  
**szerokość: 545 pikseli**  
**wysokość: 629 pikseli**




*headsstrip.png* **szerokość: 3670 pikseli, wysokość: 172 piksele**




*eyessstrip.png* **szerokość: 3670 pikseli, wysokość: 79 pikseli**




*nosesstrip.png* **szerokość: 3670 pikseli, wysokość: 86 pikseli**




*mouthsstrip.png* **szerokość: 3670 pikseli, wysokość: 117 pikseli**




*lightning\_01.jpg*



*lightning\_02.jpg*



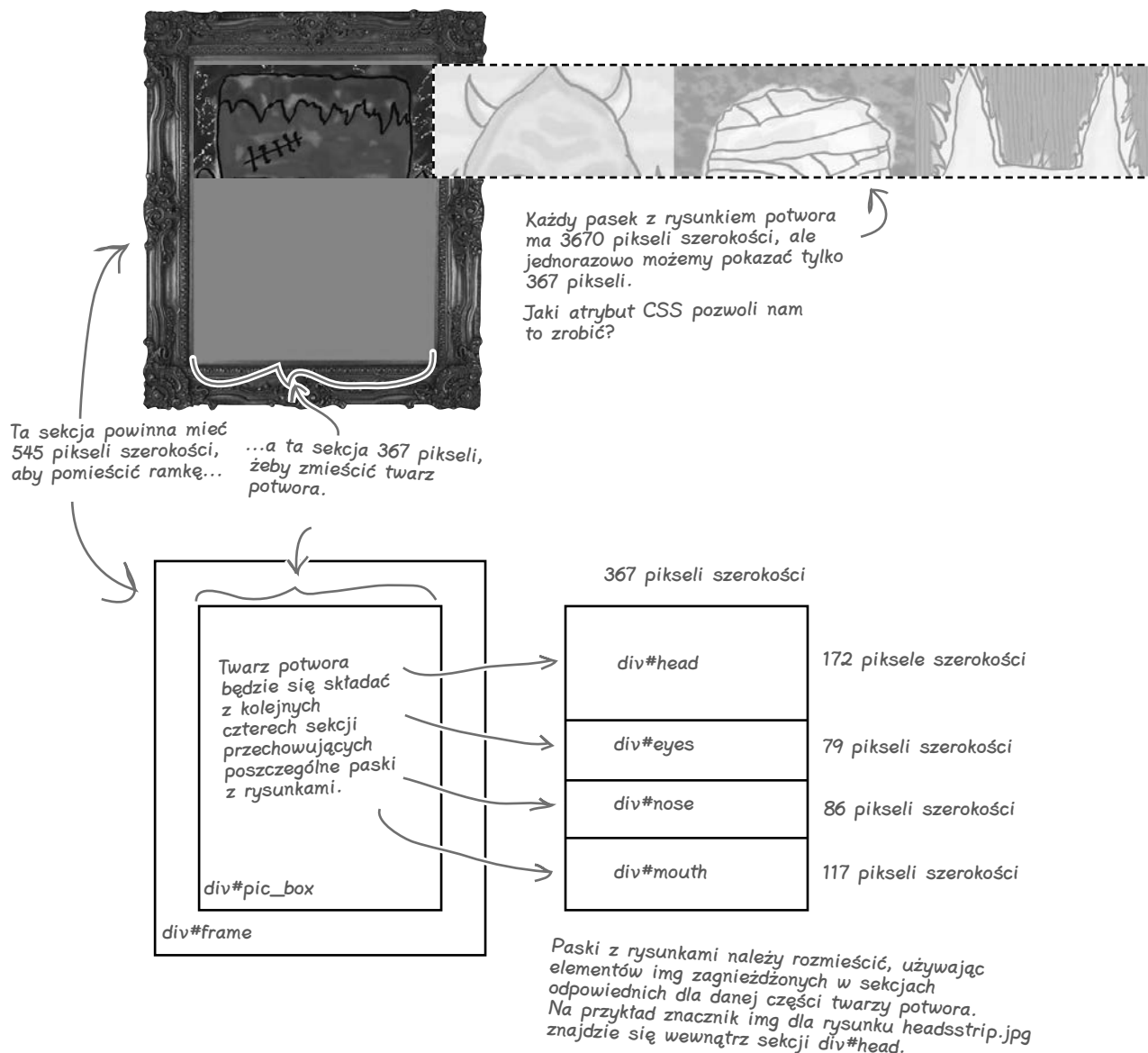
*lightning\_03.jpg*



Masz mnóstwo szczegółów dotyczących wymagań projektu oraz potrzebne pliki graficzne, ale grafik nie napisał żadnego HTML-a ani CSS, a od tego właśnie musisz zacząć. Co powinieneś zrobić, aby utworzyć i skonfigurować te pliki?

## Zrób sobie potwora — potrzebujemy układu i położenia na stronie

Bez wątpienia sporo mówiliśmy do tej pory na temat konieczności zdefiniowania już na samym początku struktury i stylu — jeszcze zanim napiszesz cokolwiek w jQuery. Teraz jest to jeszcze ważniejsze: jeżeli z góry nie określisz układu i położenia elementów na stronie, Twoje efekty oraz animacje mogą okazać się nieudane, i to *szybko*. Nie ma nic gorszego, niż gapić się w swój kod jQuery i zastanawiać się, dlaczego nie robi on w przeglądarce tego, co chciałeś. Dobrym rozwiązaniem jest naszkicowanie swoich pomysłów i przemyślenie tego, co się będzie dziać na ekranie.





Ćwiczenie

W każde z pustych miejsc w plikach HTML i CSS wpisz identyfikator CSS, właściwość albo ustawienie, które pomogą Ci skonfigurować układ i położenie elementów aplikacji Zrób sobie potwora.  
 W przypadku wątpliwości poszukaj wskazówek na dwóch poprzednich stronach. Kilka z pustych miejsc wypełniłmy za Ciebie.

```
<body>
<header id="top">
<p>Zrób sobie twarz potwora, klikając obrazek.</p></header>

<div id="frame">
  <div id="pic_box" >
    <div id=..... class="face"></div>
    <div id=..... class="face"></div>
    <div id=..... class="face"></div>
    <div id=..... class="face"></div>
  </div>
</div>
<script type="text/javascript" src="scripts/jquery-1.7.min.js"></script>
<script type="text/javascript" src="scripts/my_scripts.js"></script>
</body>
```



index.html

```
#frame {
  position: .....
  left:100px;
  top:100px;
  width:545px;
  height:629px;
  background-image:url(images/frame.png);
  z-index: 2;
  overflow: .....
}

#pic_box{
  position: relative;
  left:91px;
  top:84px;
  .....
  height:460px;
  z-index: 1;
  overflow: .....
}

.face{
  position: .....
  left:0px;
  top:0px;
  z-index: 0;
}

#head{
  height:172px;
}

#eyes{
  .....
}

#nose{
  .....
}

#mouth{
  .....
}
```



my\_style.css

## rozwiązanie ćwiczenia



### Rozwiązanie ćwiczenia

W każde z pustych miejsc w plikach HTML i CSS wpisz identyfikator CSS, właściwość albo ustawienie, które pomogą Ci skonfigurować układ i położenie elementów aplikacji Zrób sobie potwora.

W przypadku wątpliwości poszukaj wskazówek na dwóch poprzednich stronach. Kilka z pustych miejsc wypełniliśmy za Ciebie.

```
<body>
<header id="top">
<p>Zrób sobie twarz potwora, klikając obrazek.</p></header>

<div id="frame">
  <div id="pic-box">
    <div id="head" class="face"></div>
    <div id="eyes" class="face"></div>
    <div id="nose" class="face"></div>
    <div id="mouth" class="face"></div>
  </div>
</div>
<script type="text/javascript" src="scripts/jquery-1.7.min.js"></script>
<script type="text/javascript" src="scripts/my_scripts.js"></script>
</body>
```



index.html

```
#frame {
  position: absolute;
  left: 100px;
  top: 100px;
  width: 545px;
  height: 629px;
  background-image: url(images/frame.png);
  z-index: 2;
  overflow: hidden;
}

#pic_box {
  position: relative;
  left: 91px;
  top: 84px;
  width: 367px;
  height: 460px;
  z-index: 1;
  overflow: hidden;
}

.face {
  position: relative;
  left: 0px;
  top: 0px;
  z-index: 0;
}

#head {
  height: 172px;
}

#eyes {
  height: 79px;
}

#nose {
  height: 86px;
}

#mouth {
  height: 117px;
}
```

Kiedy animujemy pozycję elementów, powinniśmy używać położenia absolutnego albo względnego.

Nadanie właściwości overflow wartości „hidden” umożliwi nam ukrycie części paska z rysunkiem, która wykracza poza obszar sekcji pic\_box.

W tym celu można skorzystać też z właściwości CSS „clip”.



my\_style.css

## Trochę więcej struktury i stylu

W następnej kolejności czekają nas zmiany strukturalne w plikach HTML i CSS. Do plików *index.html* oraz *my\_style.css* dodaj poniższy kod. Pliki graficzne możesz pobrać z <ftp://ftp.helion.pl/przyklady/jquery.zip>.



*Dodaj kontener i zagnieźdź w nim rysunki z błyskawicą.*

```
<div id="container">
  
  
  
  <div id="frame">
    <div id="pic_box">
      <div id="head" class="face"></div>
      <div id="eyes" class="face"></div>
      <div id="nose" class="face"></div>
      <div id="mouth" class="face"></div>
    </div>
  </div>
</div>
```



index.html

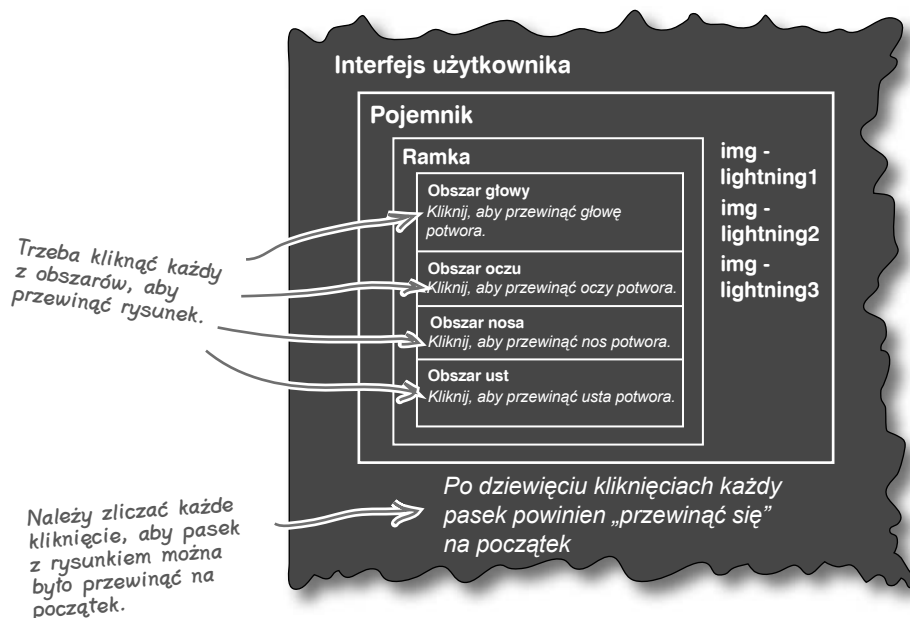
<pre><code>#container{   position: absolute;   left: 0px;   top: 0px;   z-index: 0; }  .lightning{   display: none;   position: absolute;   left: 0px;   top: 0px;   z-index: 0; }</code></pre>	<p><i>Chcemy, żeby rysunki z błyskawicą były początkowo niewidoczne.</i></p> <p><i>Kiedy chcemy animować elementy, powinniśmy nadać ich właściwości position wartość absolute, fixed albo relative.</i></p>	<pre><code>body{   background-color: #000000; }  p{   color: #33FF66;   font-family: Tahoma, Verdana, Arial, Helvetica, sans-serif;   font-size: 12px; }  #text_top {   position: relative;   z-index: 4; }</code></pre>
---	---	--



my\_style.css

## Uaktywniamy interfejs

Teraz, kiedy opracowaliśmy już wizualnie układ strony Zrób sobie potwora, skonfigurujemy resztę sekcji interfejsu użytkownika zgodnie z otrzymanym planem. Część ta sprowadza się do klikania w celu wywołania zdarzeń. Zajmujesz się tym już od czterech rozdziałów, więc konfiguracja powinna Ci pójść jak z płatka.



### Nie istnieją głupie pytania

**P:** Nie za bardzo już pamiętam, o co chodzi z tym pozycjonowaniem w CSS. Dlaczego jest nam to potrzebne do efektów i animacji w jQuery?

**U:** `position` to właściwość CSS, która kontroluje, jak i gdzie umieści elementy silnik przeglądarki odpowiedzialny za układ strony. jQuery realizuje wiele ze swoich efektów, korzystając z właściwości `position`. Jeśli już trochę pozapominałeś i potrzebujesz odświeżenia wiadomości, zajrzyj do znakomitych wyjaśnień znajdujących się w centrum deweloperskim Mozilli:

[http://developer.mozilla.org/en/CSS/position#Relative\\_positioning](http://developer.mozilla.org/en/CSS/position#Relative_positioning)

**P:** Dlaczego musimy właściwości CSS `position` nadawać wartość `absolute`, `fixed` albo `relative`, kiedy chcemy animować elementy?

**U:** Jeśli pozostawimy właściwości CSS `position` jej domyślną wartość (czyli `static`), nie będziemy mogli zastosować położenia górnego, prawego, lewego ani dolnego. Kiedy zaczniemy pracować z funkcją `animate`, będziemy potrzebowali ustawić takie położenia, a wartość `static` po prostu na to nie

pozwała. Pozostałe ustawienia właściwości `position` — `absolute`, `fixed` oraz `relative` — to umożliwiają.

**P:** Wspomniałeś o silniku przeglądarki odpowiedzialnym za układ strony. Co to takiego?

**U:** Silnik przeglądarki to jej centralna część, która interpretuje kod HTML i CSS i wyświetla go w oknie roboczym przeglądarki (czyli oknie, które pokazuje użytkownikowi zawartość strony). Google Chrome i Safari korzystają z silnika Webkit, Firefox używa silnika Gecko, a Microsoft Internet Explorer silnika o nazwie Trident.





## Magnesiki z kodem jQuery

Ułóż magnesiki z kodem w odpowiednim porządku, aby otrzymać aktywny element `div#head`. Upewnij się, że zmienne oraz instrukcje warunkowe zachowują właściwą kolejność, dzięki czemu będziesz mógł wykryć osiągnięcie przez zmienną `headclick` i x wartości 9, która oznacza dziesiąte kliknięcie.

Diagram illustrating the assembly of a jQuery code snippet for a click event on `#head`. The code is composed of several fragments arranged in a sequence:

```

var headclick
  = 0;
  if (headclick
    headclick
    = 0;
    else {
      }
    }
  }
  += 1;
  $(document).ready(function(){
    $("#head").click(function(){
      < 9){
    }
  });
  }
  });

```

The fragments are:

- `var headclick`
- `= 0;`
- `if (headclick`
- `headclick`
- `= 0;`
- `else {`
- `}`
- `});`
- `+= 1;`
- `$(document).ready(function(){`
- `$("#head").click(function(){`
- `< 9){`
- `}`
- `});`



## Magnesiki z kodem jQuery — rozwiązanie

Ułóż magnesiki z kodem w odpowiednim porządku, aby otrzymać aktywny element `div#head`. Upewnij się, że zmienne oraz instrukcje warunkowe zachowują właściwą kolejność, dzięki czemu będziesz mógł wykryć osiągnięcie przez zmienną `headcli` wartości 9, która oznacza dziesiąte kliknięcie.

```

$(document).ready(function(){
  var headcli = 0;
  $("#head").click(function(){
    if (headcli < 9){
      headcli += 1;
    }
    else {
      headcli = 0;
    }
  });
});
    
```

*Zaczynasz od 0, ponieważ jeszcze nic nie zostało kliknięte.*

*Ten warunek ogranicza użytkownika do dziesięciu kliknięć.*

*W to miejsce trafi kod animujący.*

*Nadaj zmiennej headcli wartość, jaką miała wcześniej, plus jeden.*

*Jeżeli zmienna headcli jest większa lub równa 9, zrób to.*

*W tym miejscu umieścisz kod przewijający pasek z rysunkiem.*

*Po dziesiątym kliknięciu zresetuj zmienną headcli.*

*Czy moglibyśmy w jakiś sposób powtórnie wykorzystać ten kod, żeby pozostałe elementy też były aktywne?*

### Oczywiście!

Każdy z tych elementów zachowuje się podobnie do elementu `div#head` (należy uwzględnić tylko kilka różnic takich jak nazwy zmiennych).



## Zaostrz ołówek



Uzupełnij poniższy skrypt jQuery, aby elementy oczu, nosa i ust były aktywne. Za chwilę do każdego kliknięcia dodamy odpowiednią funkcjonalność. Upewnij się, że zmienne oraz instrukcje warunkowe zachowują właściwą kolejność, dzięki czemu będziesz mógł wykryć dziesiąte kliknięcie.

```
$(document).ready(function(){  
  
    $("#head").click(function(){  
        if (headclix < 9){  
            headclix += 1;  
        }  
        else{  
            headclix = 0;  
        }  
    });  
  
});
```

```
});
```



my\_scripts.js

Zaostrz ołówek



Rozwiązanie

Tak więc uaktywniłeś elementy oczu, nosa i ust, ustawiając w odpowiedniej kolejności zmienne oraz instrukcje warunkowe, dzięki czemu będziesz mógł wykryć dziesiąte kliknięcie.

```
$(document).ready(function(){
    var headclix = 0, eyeclix=0, noseclix= 0, mouthclix = 0;

    $("#head").click(function(){
        if (headclix < 9){
            headclix += 1;
        }
        else{
            headclix = 0;
        }
    });

    $("#eyes").click(function() {
        if (eyeclix < 9){
            eyeclix += 1;
        }
        else{
            eyeclix = 0;
        }
    });

    $("#nose").click(function() {
        if (noseclix < 9){
            noseclix += 1;
        }
        else{
            noseclix = 0;
        }
    });

    $("#mouth").click(function() {
        if (mouthclix < 9){
            mouthclix += 1;
        }
        else{
            mouthclix = 0;
        }
    });
});
```

Możemy zadeklarować wiele zmiennych i nadać im wartości, rozdzielając te zmienne przecinkami.

Każda część twarzy potwora jest teraz aktywna i skonfigurowana w taki sposób, aby przed przewinięciem paska z rysunkiem pozwolić tylko na dziewięć kliknięć.

Czy zwrócisz uwagę, że — pomimo drobnych różnic — każda funkcja wewnątrz metody click ma podobną strukturę? To mógłby być dobry przypadek do wielokrotnego wykorzystania kodu.

Cierpliwości, mój pasikoniku, dojdziemy do tego w rozdziale 7.



my\_scripts.js

## Robimy efekt błyskawicy

Następny w kolejności jest efekt błyskawicy. Zanim spróbujemy zrealizować jego działanie, sprawdźmy, co na ten temat jest w planie.

**Interfejs użytkownika**

**Pojemnik**

**Ramka**

**Obszar głowy**  
Kliknij, aby przewinąć głowę potwora.

**Obszar oczu**  
Kliknij, aby przewinąć oczy potwora.

**Obszar nosa**  
Kliknij, aby przewinąć nos potwora.


**Obszar ust**  
Kliknij, aby przewinąć usta potwora.

img - lightning1  
img - lightning2  
img - lightning3

Rysunki z błyskawicą są zagnieżdżone w sekcji pojemnika...

...i muszą się szybko pojawiać i znikać.

Symulacja obrazująca, jak powinna wyglądać animacja błyskawicy.



Obrazy błyskawicy powinny pojawiać się i znikać szybko, tak jakby rzeczywiście się błysnęło.



Zrobiliśmy już coś podobnego w rozdziale 1. z wysuwaniem się rysunku i nabieraniem przez niego barw. Czy nie moglibyśmy skorzystać z tego samego, żeby Zrób sobie potwora zadziało?

**Potencjalnie tak, ale być może istnieje lepszy sposób.**

Przyjrzelśmy się gotowym efektom jQuery w rozdziale 1., ale teraz zagłębimy się w nie trochę bardziej.

## W jaki sposób jQuery animuje elementy?

Kiedy przeglądarka pobiera plik CSS, ustawia wizualne właściwości elementów na stronie. Korzystając z wbudowanych efektów jQuery, interpreter JS zmienia te właściwości i animuje elementy na Twoich oczach. Nie jest to jednak magia... wszystko sprowadza się do właściwości CSS. Spójrzmy jeszcze raz na kilka efektów, które już widziałeś.

### hide, show i toggle zmieniają właściwość CSS display

hide



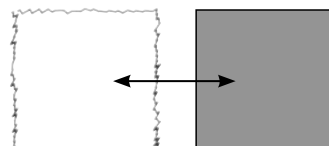
Interpreter JS zmienia właściwość CSS display wybranego elementu na none i usuwa go z układu strony.

show



Interpreter JS tak zmienia właściwość CSS display wybranego elementu, aby stał się on widoczny.

toggle



Jeśli element jest ukryty, interpreter JS go pokazuje, i odwrotnie.

Efekty jQuery zmieniają właściwości CSS w locie, wprowadzając zmiany na stronie na oczach użytkownika.



### WYSIL SZARE KOMÓRKI

hide, show i toggle zmieniają właściwość display. Potrzebujemy jednak przewijać części twarzy oraz w tym samym czasie włączać i wyłączać błyskawice. Którą właściwość CSS zmienia Twoim zdaniem jQuery w celu uzyskania tych efektów?

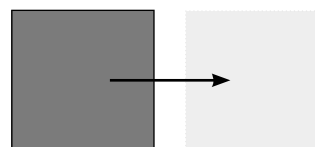
## Efekty wygaszania zmieniają właściwość CSS opac i ty

### fadeIn



Za pomocą fadeIn interpreter JavaScriptu zmienia właściwość CSS opac i ty wybranego elementu od 0 do 100.

### fadeTo



fadeTo umożliwia animowaną zmianę właściwości opac i ty wybranego elementu do określonej wartości procentowej.

### fadeOut



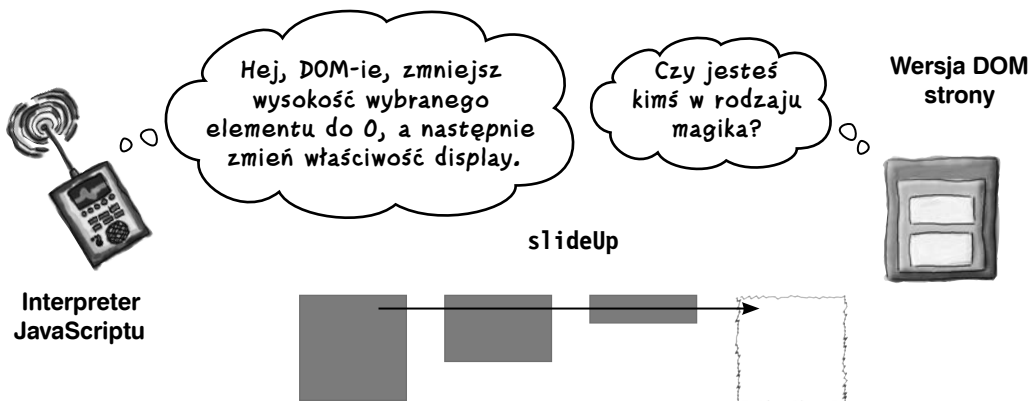
Za pomocą fadeOut interpreter JavaScriptu zmienia właściwość CSS opac i ty wybranego elementu od 100 do 0 i pozostawia na stronie miejsce dla tego elementu.



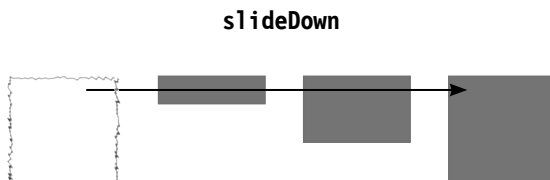
### Porady maniaaka

Właściwość CSS opac i ty nie działa tak samo na różnych przeglądarkach. Na szczęście jQuery zadba o to w naszym imieniu i tak naprawdę to wszystko, co musimy o tym wiedzieć!

## Wysuwanie sprowadza się do zmiany wysokości



Interpreter JavaScriptu nakazuje drzewu DOM zmienić właściwość CSS `height` wybranego elementu (lub elementów) na 0, a następnie nadać właściwości `display` wartość `none`. W zasadzie jest to ukrycie za pomocą wsunięcia.



Interpreter JavaScriptu pokazuje wybrany element (lub elementy), animując ich wysokość od 0 do wartości ustawionej w stylu CSS.



Interpreter JavaScriptu sprawdza, czy obraz ma pełną wysokość, czy zerową, i przełącza efekt wysunięcia w zależności od wyniku. Jeżeli element ma wysokość 0, interpreter wysuwa go w dół, a jeśli ma pełną wysokość, wsuwa go w górę.





Mogę więc wysuwać elementy wyłącznie w górę i w dół? A co, jeśli bym chciała przesunąć coś w lewo albo w prawo?

**jQuery udostępnia gotowe efekty tylko do wysuwania elementów w górę albo w dół.**

W bibliotece tej nie znajdziesz metody `slideRight` ani `slideLeft` (a przynajmniej nie było ich w chwili pisania tej książki). Nie przejmuj się jednak — zajmiemy się tym nieco później...

W jQuery nie znajdziesz metody `slideRight` ani `slideLeft`.

**Zaostrz ołówek**



Które z gotowych efektów jQuery przydadzą się w aplikacji Zrób sobie potwora? Dla każdej grupy efektów napisz, czy będą one pomocne, i wyjaśnij, dlaczego tę grupę wybrałeś lub nie.

Efekt	Czy możemy go użyć?	Dlaczego?
Pokazywanie/ ukrywanie		
Wysunięcia		
Wygaszania		

## Zaostrz ołówek



### Rozwiązanie

Które z gotowych efektów jQuery przydadzą się w aplikacji Zrób sobie potwora?

Efekt	Czy możemy go użyć?	Dlaczego?
Pokazywanie/ukrywanie	Nie	Efekty pokazywania i ukrywania ( <i>show</i> i <i>hide</i> ) nie pomogą nam przy tworzeniu aplikacji Zrób sobie potwora, ponieważ nie musimy animować właściwości <i>display</i> żadnego elementu.
Wysunięcia	Nie	O mały włoś! Pasek z rysunkiem musimy przesunąć w lewo, a <i>slideUp</i> , <i>slideDown</i> i efekty wysunięć pozwalają na zmienianie tylko właściwości <i>height</i> . Potrzebujemy czegoś, co zmieniłoby właściwość <i>left</i> .
Wygaszenia	Tak	Możemy skorzystać z efektów wygaszania, aby sprostać wymogom planu, które stwierdzają, że rysunki błyskawic powinny pojawiać się i wygaszać szybko, jakby naprawdę się błyskało.

## Zatrudnij efekty wygaszania

W założeniach jest mowa o tym, że rysunki błyskawic powinny pojawiać się i wygaszać, ale ma się to dziać szybko, aby wyglądało tak, jakby naprawdę się błyskało. Zagłębmy się trochę w efekty wygaszania i zobaczmy, jak możemy sprawić, że błyskanie będzie działać.

Tutaj jest identyfikator pierwszego elementu graficznego.

Tutaj wstawiamy parametr *duration* decydujący o czasie trwania animacji. Kontroluje on to, jak długo będzie trwał efekt.

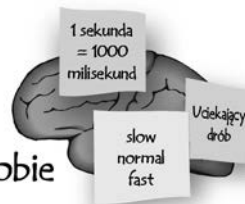
```
$("#lightning1").fadeIn("fast");
```

Możesz użyć jednego z parametrów tekstowych: *slow*, *normal* lub *fast*...

...albo możesz podać wartość w milisekundach. Mógłbyś na przykład wstawić 1000 i animacja efektu trwałaby jedną sekundę.

```
$("#lightning1").fadeIn(1000);
```

Przyklej to sobie



## Łącz efekty za pomocą łańcuchów metod

Błyskawice muszą pojawiać się i wygaszać wciąż na nowo. Zamiast rozpisywać te efekty oddzielnie, możemy skorzystać z łańcucha metod, z którego korzystaliśmy przez chwilę w rozdziale 4., kiedy musieliśmy wspinać się po drzewie DOM. Łańcuchy metod to cecha jQuery pozwalająca na łączenie ze sobą metod, które będą działać na zwróconym zbiorze elementów. Dzięki nim efekty błyskawic będą łatwiejsze do napisania, a zatem przyjrzyjmy się im bliżej.

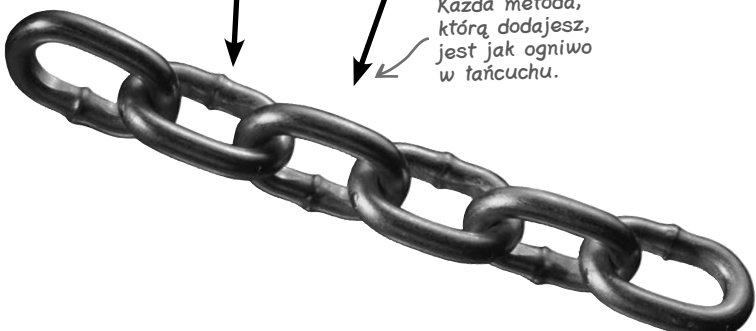
Właściwość display elementu zmieni się od ukrycia do pełnej widoczności i nieprzezroczystości...

...a następnie element znowu zostanie zmieniony w całkowicie przezroczysty.

Jeśli między nawiasami nie określisz czasu trwania, efekt otrzyma domyślną wartość normal, która wynosi 400 milisekund, czyli 0,4 sekundy.

```
$("#lightning1").fadeIn().fadeOut();
```

Każda metoda, którą dodajesz, jest jak ogniwo w łańcuchu.




### Ćwiczenie

Napisz wiersze kodu, które zrealizują każdy z etapów przedstawionych poniżej.

- 1 Pokaż stopniowo element #lightning1 w czasie jednej czwartej sekundy.

.....

- 2 Dodaj kolejny efekt, który wygasi element #lightning1 w czasie jednej czwartej sekundy.

.....



Rozwiązanie ćwiczenia

Napisz wiersze kodu, które zrealizują każdy z etapów przedstawionych poniżej.

- 1 Pokaż stopniowo element #lightning1 w czasie jednej czwartej sekundy.

```
$("#lightning1").fadeIn("250");
```

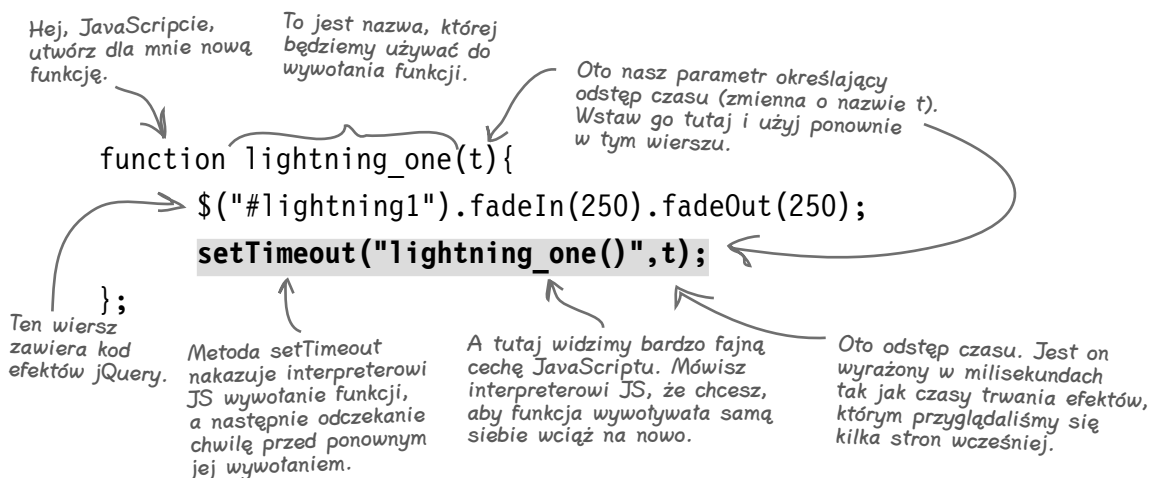
- 2 Dodaj kolejny efekt, który wygasi element #lightning1 w czasie jednej czwartej sekundy.

```
$("#lightning1").fadeIn("250").fadeOut("250");
```

## Kontratakujemy za pomocą funkcji czasowych

Masz już więc błyskawicę, która pojawia się i znika, ale wymogi projektu mówią, że błyskawice powinny uderzać przez cały czas. Prawdziwa błyskawica strzela przez niebo, po czym zwykle następuje przerwa, po której na niebie pojawia się kolejna. Potrzebujemy zatem *powtarzać* nasz efekt.

Powróć myślami do poprzednich rozdziałów, gdzie musiałeś wykonywać powtarzalne zadania. Z czego korzystałeś? Zgadza się: z funkcji! Po raz pierwszy pojawiły się one w rozdziale 3., gdy tworzyliśmy funkcję wielokrotnego użytku obsługującą kliknięcia oraz generator liczb losowych. Teraz możemy użyć funkcji do wywoływania wygaszeń, poczekać chwilę, a następnie znowu je wywoływać w różnych odstępach czasu. Dzięki temu uzyskamy dla Zrób sobie potwora udany efekt wielu błyskawic. Spójrzmy na funkcję realizującą taki efekt.



**W zaledwie trzech wierszach kodu uzyskałeś zsynchronizowaną w czasie funkcję błyskającą pierwszym rysunkiem pioruna. Spróbuj teraz napisać funkcje dla pozostałych dwóch rysunków z piorunami.**



## Magnesiki jQuery

Poukładaj magnesiki z kodem we właściwej kolejności, aby uzyskać funkcje animujące błyskawice dla pozostałych dwóch elementów z piorunami.

```

function lightning_two (t){
};

function lightning_three (t){
};

$("#lightning2").fadeIn(250);
setTimeout("lightning_two()", t);
$("#lightning2").fadeOut(250);

$("#lightning3").fadeIn(250);
setTimeout("lightning_three()", t);
$("#lightning3").fadeOut(250);
    
```



## Magnesiki jQuery — rozwiązanie

Poukładaj magnesiki z kodem we właściwej kolejności, aby uzyskać funkcje animujące błyskawice dla pozostałych dwóch elementów z piorunami.

```
function lightning_two (t){
  $("#lightning2") .fadeIn(250) .fadeOut(250);
  setTimeout("lightning_two()", t);
};

function lightning_three (t){
  $("#lightning3") .fadeIn(250) .fadeOut(250);
  setTimeout("lightning_three()", t);
};
```

Nie istnieją  
głupie pytania

**P:** Czy `fadeIn().fadeOut()` nie jest tym samym co `toggle`?

**O:** Dobre pytanie! To nie jest to samo. Metoda `toggle` jest pojedynczą metodą, która po prostu przełącza wybrany element ze stanu ukrytego na widoczny i na odwrót w zależności od bieżącego stanu tego elementu. Połączenie `fadeIn` i `fadeOut` w łańcuch utworzy sekwencyjny efekt, który najpierw stopniowo pokaże wybrany element lub elementy, po czym — gdy to zostanie już osiągnięte — ponownie je wygasi.

**P:** Metoda `setTimeout` jest dla mnie nowa. Czy pochodzi ona z jQuery, czy z JavaScriptu?

**O:** Metoda `setTimeout` pochodzi tak naprawdę z JavaScriptu i możesz z niej korzystać w celu kontrolowania pewnych aspektów animacji w jQuery. Funkcją `setTimeout` zajmiemy się dokładniej w dalszych rozdziałach, a zwłaszcza w rozdziale 7.

Jeśli chciałbyś poczytać o niej już teraz, odwiedź Mozilla Developer's Center pod adresem <https://developer.mozilla.org/en/window.setTimeout>, a jeżeli

naprawdę chcesz pogłębić swoją wiedzę, sięgnij po znakomitą książkę Davida Flanagana *JavaScript: The Definitive Guide* (O'Reilly; <http://oreilly.com/catalog/9780596805531>).

**P:** Kiedy używam efektu `hide`, element po prostu znika. Jak mogę to spowolnić?

**O:** Aby spowolnić efekt `hide`, `show` albo `toggle`, dodaj między nawiasami parametr `duration` określający czas trwania animacji. Oto, jak moglibyśmy zrealizować efekt ukrywania w rozdziale 1.: `$("#picframe").hide(500);`

## Dodaj do swojego skryptu funkcje błyskawic

Korzystając z kodu, który utworzyłeś w ćwiczeniu na poprzedniej stronie, zaktualizuj plik ze skryptem dla aplikacji Zrób sobie potwora.

Zrób to!

W tych wierszach są wywoływane funkcje zdefiniowane na samym dole pogrubioną czcionką.

```
$(document).ready(function(){
    var headclix = 0, eyeclix = 0, noseclix = 0, mouthclix = 0;
    lightning_one(4000);
    lightning_two(5000);
    lightning_three(7000);
    $("#head").click(function(){
        if (headclix < 9){headclix+=1;}
        else{headclix = 0;}
    });
    $("#eyes").click(function(){
        if (eyeclix < 9){eyeclix+=1;}
        else{eyeclix = 0;}
    });
    $("#nose").click(function(){
        if (noseclix < 9){noseclix+=1;}
        else{noseclix = 0;}
    });
    $("#mouth").click(function(){
        if (mouthclix < 9){mouthclix+=1;}
        else{mouthclix = 0;}
    });
});//koniec funkcji end doc.onready
```

Liczby między nawiasami to parametry w milisekundach, które zostaną przekazane do metody setTimeout. Za ich pomocą możesz zmieniać czasy błyskania piorunów.

Dla oszczędności miejsca usunęliśmy niektóre podziały wierszy. Nie przejmuj się, jeśli w Twoim skrypcie wyglądają one inaczej.

To są definicje funkcji błyskawic.

```
function lightning_one(t){
    $("#container #lightning1").fadeIn(250).fadeOut(250);
    setTimeout("lightning_one()",t);
};
function lightning_two(t){
    $("#container #lightning2").fadeIn("fast").fadeOut("fast");
    setTimeout("lightning_two()",t);
};
function lightning_three(t){
    $("#container #lightning3").fadeIn("fast").fadeOut("fast");
    setTimeout("lightning_three()",t);
};
```



my\_scripts.js



# Jazda próbna

Otwórz stronę w swojej ulubionej przeglądarce, aby sprawdzić, czy Twój efekt błyskawicy działa.



Udało Ci się osiągnąć efekt wygaszania błyskawic, łącznie go z metodą JavaScriptu `setTimeout`.

Pioruny pojawiają się i znikają szybko w różnych odstępach czasu, symulując prawdziwe błyskawice.

Jak dotąd udało Ci się opracować funkcje obsługujące kliknięcia, a trzy rysunki błyskawic pojawiają się i wygaszają w różnych odstępach czasu. Spójrzmy na plan, aby sprawdzić, co pozostało jeszcze do zrobienia.

## Projekt Zrób sobie potwora

Zadaniem aplikacji Zrób sobie potwora jest zabawianie dzieci w docelowej grupie wiekowej przez umożliwienie im zrobienia sobie potwora dzięki mieszaniu 10 różnych głów, oczu, nosów i ust. Przejścia między poszczególnymi częściami twarzy potwora powinny być animowane.

### Animacja

Symulacja obrazująca, jak powinna się zmieniać część twarzy potwora.



Symulacja obrazująca, jak powinna wyglądać animacja błyskawicy.

Ta część planu stanowi nasze ostatnie wyzwanie w projekcie.

Dotarliśmy więc do miejsca, w którym potrzebujemy przesunąć rysunek w lewo, a żaden z gotowych efektów przesuwania tego nie robi. Czy istnieje jakaś inna metoda, której moglibyśmy użyć?

**Gotowe efekty są wspaniałe, ale nie pozwalają Ci robić wszystkiego, na co tylko masz ochotę.**

Nadeszła pora na opracowanie niestandardowego efektu, który będzie przesuwać części twarzy potwora w lewo.





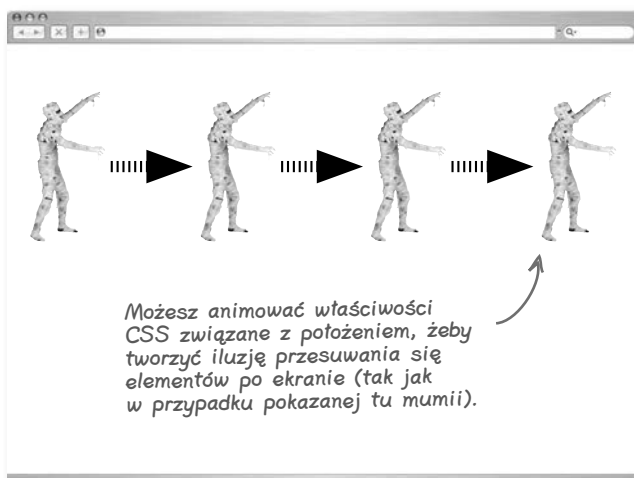
## Efekty własnej roboty wykorzystujące metodę animate

Tak więc w jQuery nie ma efektu `slideRight` ani `slideLeft`, a dokładnie tego potrzebujesz na tym etapie realizacji projektu. Czy to oznacza, że nasza strona Zrób sobie potwora poległa?

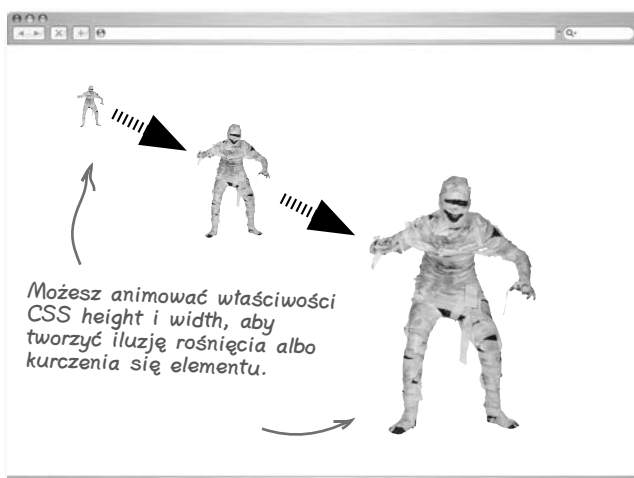
Bez obaw. jQuery oferuje metodę `animate` służącą do tworzenia własnych efektów. Za pomocą `animate` możesz tworzyć niestandardowe animacje, które będą potrafiły robić o wiele więcej niż gotowe efekty. Metoda ta umożliwi animowanie właściwości CSS wybranego elementu lub elementów, a także animowanie wielu właściwości w tym samym czasie.

Rzucmy okiem na niektóre z efektów, jakie możesz osiągnąć za pomocą metody `animate`.

### Efekty ruchu



### Efekty skali



## WYSIL SZARE KOMÓRKI

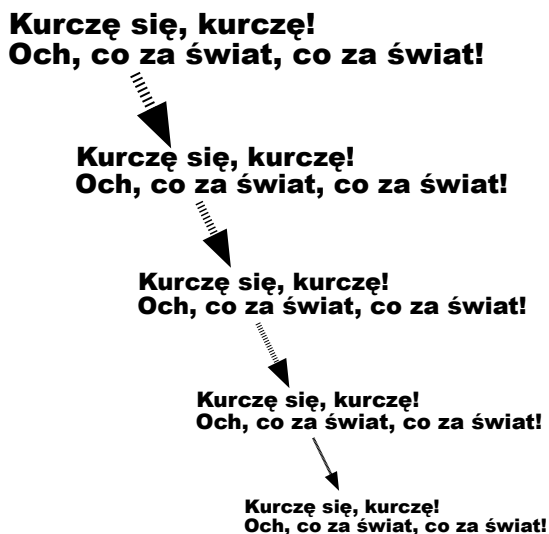
Którą właściwość CSS będziesz musiał animować, aby przy każdym kliknięciu części twarzy potwora przesunął się w lewo?

## Co można, a czego nie można animować?

Za pomocą metody `animate` możesz dynamicznie zmieniać właściwości czcionek, aby tworzyć efekty tekstowe. W jednym wywołaniu animacji możesz także animować wiele właściwości CSS jednocześnie, co poszerza paletę ciekawych rzeczy, które potrafi robić Twoja aplikacja sieciowa.

Chociaż metoda `animate` jest naprawdę niezła, ma ona swoje ograniczenia. Gdzieś tam w głębi animacja wykorzystuje mnóstwo matematyki (czym na szczęście nie musisz się przejmować), a zatem jesteś ograniczony do pracy tylko z tymi właściwościami CSS, których ustawienia są *numeryczne*. Znaj swoje ograniczenia, ale popuść wodze wyobraźni — metoda `animate` oferuje pełne spektrum elastyczności i rozrywki.

### Efekty tekstowe



Możesz animować właściwości CSS czcionek, aby tworzyć iluzję lecącego, rosnącego albo malejącego tekstu.

To zaledwie kilka przykładów. Potrzebowalibyśmy o wiele, wiele, wiele więcej stron w książce, żeby zaprezentować wszystkie możliwości.



Obejrzyj to!

#### Metoda `animate` będzie działać wyłącznie z właściwościami CSS, które w swoich ustawieniach korzystają z liczb:

- obramowanie, margines, dopełnienie;
- wysokość elementu, wysokość minimalna i maksymalna;
- szerokość elementu, szerokość minimalna i maksymalna;
- rozmiar czcionki;
- pozycja dolna, lewa, prawa i górna;
- pozycja tła;
- odstępy między literami i wyrazami;
- wcięcie tekstu;
- wysokość linii.



## Metoda animate z bliska

Z zewnątrz metoda animate działa tak jak inne metody, z którymi już pracowałeś.

Wybierz element lub elementy, z którymi chcesz pracować.

Wywołaj metodę animate.

Pierwszy parametr pozwala Ci wybrać właściwość CSS, którą chcesz animować.

Drugi parametr to czas trwania w milisekundach. Pozwala Ci on określić, jak długo będzie trwać animacja.

```
$("#my_div").animate({left:"100px"},500);
```

W tym przykładzie animujemy właściwość CSS left...

...i ustawiamy ją na 100 pikseli.

Pierwszy parametr jest wymagany — musisz go tu umieścić, aby animacja zadziałała. Drugi parametr jest opcjonalny.

Jedną z największych zalet metody animate jest jednak możliwość zmieniania wielu właściwości wybranego elementu w tym samym czasie.

```
$("#my_div").animate({
  opacity: 0,
  width: "200",
  height: "800"
}, 5000);
```

W tym przykładzie animujemy jednocześnie przezroczystość i wielkość elementu.



**Parametry właściwości CSS należy ustawiać zgodnie ze standardem DOM, a nie ze standardem CSS.**



## WYTEŻ UMYSŁ

Jak myślisz — co takiego dzieje się za kulisami w przeglądarce, co pozwala metodzie animate zmieniać elementy na oczach użytkownika?

## Metoda animate zmienia w czasie styl

Efekty wizualne oraz animacja, które oglądasz na ekranie w kinie albo w telewizji, korzystają ze złudzenia ruchu. Specjaliści od efektów i animatorzy biorą sekwencję obrazów i odtwarzają je w określonym tempie po jednym obrazie, aby to złudzenie osiągnąć. Spotkałeś się zapewne z opracowanymi tym prostym sposobem książkami, w których ten sam efekt jest osiągnięty podczas ich wertowania.

To samo dzieje się w oknie przeglądarki, z tym że nie mamy sekwencji obrazów do pokazania. Zamiast tego interpreter JavaScriptu wielokrotnie wywołuje funkcję, która zmienia styl animowanego obiektu. Przeglądarka przenosi te zmiany na ekran. Użytkownik widzi iluzję ruchu albo zmianę elementu, kiedy modyfikowany jest jego styl.

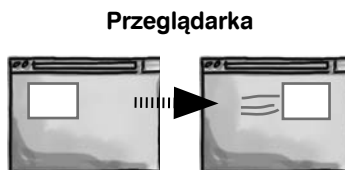
- 1 Kiedy działa metoda animate, interpreter JavaScriptu ustawia licznik na czas trwania animacji.



- 2 Interpreter JavaScriptu mówi silnikowi przeglądarki, żeby zmienił właściwość CSS określoną w parametrze metody animate. Silnik przeglądarki wyświetla tę właściwość na ekranie.



- 3 Interpreter JavaScriptu wielokrotnie wywołuje funkcję zmieniającą właściwość CSS elementu do chwili, w której licznik ustawiony w punkcie 1 wyzeruje się. Przy każdym wywołaniu funkcji zmiana jest pokazywana na ekranie.
- 4 Użytkownik widzi iluzję ruchu, kiedy przeglądarka renderuje zmiany elementu.



## \* \* ? \* KTO CO ROBI?

Połącz fragmenty kodu zawierające metodę animate z opisem tego, co robi on na ekranie.

```
$("#my_div").animate({top: "150px"}, "slow")
```

Jednocześnie animuje zmiany lewego i prawego marginesu wszystkich paragrafów.

```
$("#p").animate({
  marginLeft:"150px",
  marginRight:"150px"
});
```

Animuje zmianę prawej pozycji #my\_div do zera w ciągu pół sekundy.

```
$("#my_div").animate({width: "30%"}, 250)
```

Animuje zmianę odstępu między literami we wszystkich sekcjach w domyślnym czasie 400 milisekund.

```
$("#my_div").animate({right: "0"}, 500)
```

Jednocześnie animuje zmianę dopełnienia i szerokości #my\_div.

```
$("#p").animate({letterSpacing:"15px"});
```

Powoli animuje zmianę górnej pozycji #my\_div.

```
$("#my_div").animate({
  padding: "200px",
  width: "30%"
}, "slow")
```

Szybko animuje zmianę wysokości wszystkich obrazów.

```
$("#img").animate({height: "20px"}, "fast")
```

Animuje zmianę szerokości #my\_div w czasie jednej czwartej sekundy.

# \* KTO CO ROBI? \*

## Rozwiązanie

Połącz fragmenty kodu zawierające metodę animate z opisem tego, co robi on na ekranie.

```
$("#my_div").animate({top: "150px"}, "slow")
```

Jednocześnie animuje zmiany lewego i prawego marginesu wszystkich paragrafów.

```
$("#p").animate({  
  marginLeft: "150px",  
  marginRight: "150px"  
});
```

Animuje zmianę prawej pozycji #my\_div do zera w ciągu pół sekundy.

```
$("#my_div").animate({width: "30%"}, 250)
```

Animuje zmianę odstępu między literami we wszystkich sekcjach w domyślnym czasie 400 milisekund.

```
$("#my_div").animate({right: "0"}, 500)
```

Jednocześnie animuje zmianę dopełnienia i szerokości #my\_div.

```
$("#p").animate({letterSpacing: "15px"});
```

Powoli animuje zmianę górnej pozycji #my\_div.

```
$("#my_div").animate({  
  padding: "200px",  
  width: "30%"  
}, "slow")
```

Szybko animuje zmianę wysokości wszystkich obrazów.

```
$("#img").animate({height: "20px"}, "fast")
```

Animuje zmianę szerokości #my\_div w czasie jednej czwartej sekundy.

## Dokładnie skąd dokąd?

Ważną rzeczą do zapamiętania na temat metody animate jest fakt, że zmienia ona *aktualną* właściwość CSS na właściwość, którą ustawisz w *pierwszym parametrze*. Aby Twoje niestandardowe animacje były skuteczne, musisz się mocno zastanowić, jakie wartości są aktualnie ustawione w CSS. W poprzednim przykładzie zmieniliśmy lewe położenie #my\_div na 100 pikseli. To, co wydarzy się na ekranie, w całości zależy od bieżącej wartości właściwości CSS left elementu #my\_div.

### Bieżąca wartość właściwości CSS



```
#my_div{
  left: 20px;
}
```

### Wartość właściwości CSS w animate



#my\_div przesunie się o 80 pikseli w prawo.

Element jest animowany do położenia bezwzględnego.

```
$("#my_div").animate({left:"100px"});
```

Jeśli właściwość ma aktualnie inną wartość, otrzymamy inny wynik.

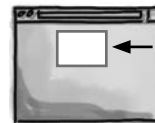
### Bieżąca wartość właściwości CSS



#my\_div zaczyna się od dwusetnego piksela

```
#my_div{
  left: 200px;
}
```

### Wartość właściwości CSS w animate



#my\_div przesunie się o 100 pikseli w lewo, ponieważ po lewej stronie znajduje się mniej pikseli.

```
$("#my_div").animate({left:"100px"});
```

Fascynujące! Tylko jak możemy z tego skorzystać, żeby zadziałało nasze Zrób sobie potwora?



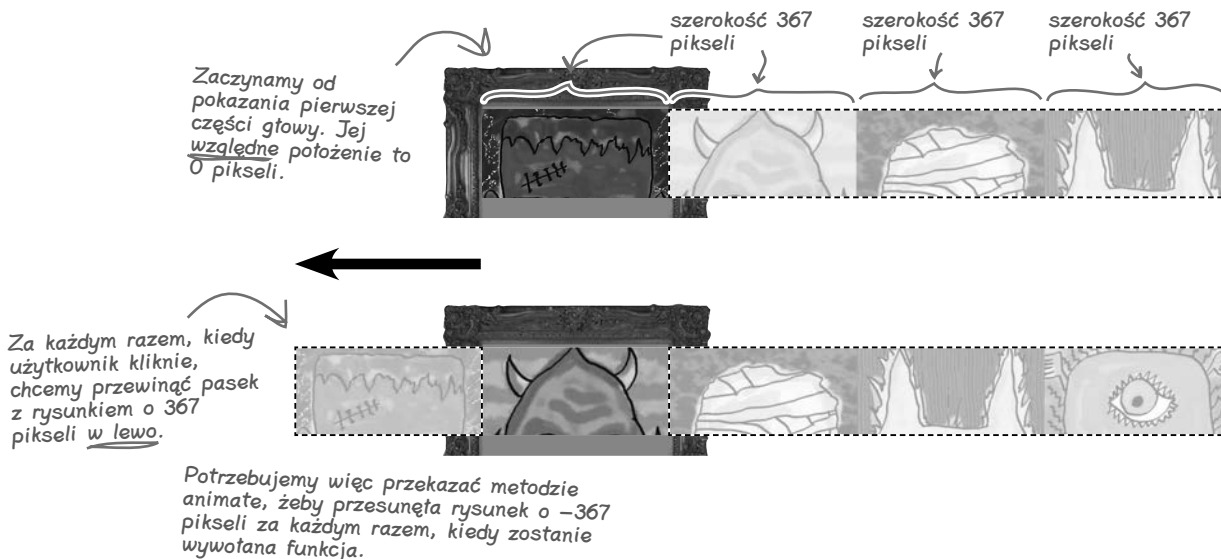
### Wszystko jest względne.

Aby części twarzy potwora w Zrób sobie potwora przesuwały się w stronę, w którą chcemy, musimy zastanowić się, jakie są ich *bieżące pozycje* i jak chcemy je zmienić względem *pozycji*, które zajęły po tym, kiedy ostatnio zmieniła je metoda animate.

to jest względnie łatwe

## Ruch bezwzględny obiektów a ich ruch względny

Zapewne pamiętasz, że paski z rysunkami, które chcemy pokazać, zagnieździłismy wewnątrz sekcji o identyfikatorze `#pic_box`. Obecnie w CSS właściwość `left` elementu `#pic_box` ma wartość `91px`. Zastanówmy się nad tym, jak chcielibyśmy przesunąć paski z rysunkami, aby uzyskać efekt przesunięcia w lewo, na którym nam zależy.



Zastanów się nad przykładem animacji bezwzględnej z poprzedniej strony.

Tutaj mówimy metodzie `animate`, żeby nadała lewej pozycji sekcji `#my_div` wartość równą dokładnie 100 pikseli.

```
$("#my_div").animate({left:"100px"});
```

Ale jak powiedzieć jej, żeby przesunęła element o `-367` pikseli za każdym razem, kiedy metoda `animate` zostanie wywołana?

```
$("#head").animate({left:"???"});
```

## Animacja względna = przesuń to za każdym razem właśnie o tyle

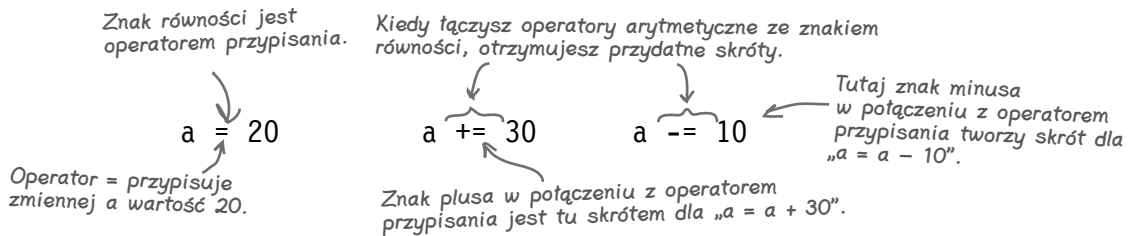
Za pomocą animacji bezwzględnej przesuwasz element na bezwzględną pozycję siatki ekranu. Za pomocą animacji względnej przesuwasz element *względem* pozycji, którą zajmował *ostatnio* po zakończeniu animacji, która go tam umieściła.

**Ale jak przesuujemy element względnie za pomocą metody `animate`?**



## Przesuwaj elementy względnie dzięki łączeniu operatorów

Istnieją specjalne operatory JavaScriptu, które przesuwają element lub elementy o tę samą wartość przy każdym wywołaniu metody `animate`. Są one znane jako *operatory przypisania*, ponieważ są zwykle używane do przypisywania wartości zmiennej w taki sposób, że do jej bieżącej wartości jest dodawana nowa wartość. Brzmi to o wiele bardziej skomplikowanie, niż w istocie jest.



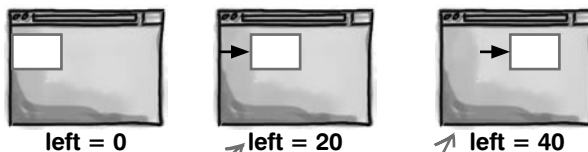
Takie połączenia operatorów pomagają w tworzeniu animacji względnej, umożliwiając ustawianie nowej wartości na wartość bieżącą *plus* albo *minus* pewna liczba pikseli.

Ten zapis przesunie element o identyfikatorze `box` o 20 pikseli za każdym razem, kiedy zostanie wywołana metoda `animate`.

```
$("#box").animate({left:"+=20"});
```

Oto, co się stanie z elementem `#box` przy każdym wywołaniu powyższej metody `animate`.

Załóżmy, że `left` startuje z wartością 0.



Zwiększając za każdym razem wartość lewej pozycji elementu, w rzeczywistości przesuwamy go w prawą stronę okna przeglądarki.



### CELNE SPOSTRZEŻENIA

Oto dwa inne zestawienia operatorów przypisania:

- `a *= 5` jest skrótem dla przemnożenia bieżącej wartości `a` przez 5 i przypisania uzyskanego wyniku do `a`.
- `a /= 2` jest skrótem dla podzielenia bieżącej wartości `a` przez 2 i przypisania uzyskanego wyniku do `a`.



### Ćwiczenie

Napisz wiersz kodu w jQuery, który zrealizuje następujące czynności:

- 1 Przesunięcie elementu `#head` o 367 pikseli w lewo przy każdym wywołaniu metody `animate`. Czas trwania ustaw na pół sekundy.
- 2 Przesunięcie elementu `#head` na pozycję wyjściową (`left:0px`). Czas trwania ustaw na pół sekundy.



### Rozwiązanie ćwiczenia

Napisz wiersz kodu w jQuery, który zrealizuje następujące czynności:

- 1 Przesunięcie elementu #head o 367 pikseli w lewo przy każdym wywołaniu metody `animate`. Czas trwania ustaw na pół sekundy.

```
$("#head").animate({left:"-367px"},500);
```

- 2 Przesunięcie elementu #head na pozycję wyjściową (`left:0px`). Czas trwania ustaw na pół sekundy.

```
$("#head").animate({left:"0px"},500);
```

← Ta bezwzględna animacja resetuje głowę potwora w sposób przypominający przewinięcie taśmy.

## Nie istnieją głupie pytania

**P:** Niektóre osoby nie chcą, żeby animacje przeszkadzały im w oglądaniu strony. Co powinienem zrobić, żeby pozwolić użytkownikowi na wyłączenie animacji?

**U:** To doskonałe spostrzeżenie. Animacje mogą irytować i powodować problemy z dostępnością. Jeśli chcesz, żeby użytkownicy mogli je wyłączać, możesz utworzyć przycisk powiązany z następującym kodem (wiesz już, jak to zrobić):

```
$.fx.off = true;
```

Innym przydatnym sposobem na wyłączenie animacji jest użycie metody jQuery o nazwie `stop`. Możesz się dowiedzieć więcej na temat obu tych sposobów w witrynie jQuery:

<http://api.jquery.com/jquery.fx.off/>

<http://api.jquery.com/stop/>

**P:** Powiedziałeś: „Parametry właściwości CSS należy ustawiać zgodnie ze standardem DOM, a nie ze standardem CSS”. Co to do diaska znaczy?

**U:** Świetne pytanie! Metoda `animate` pobiera parametry napisane w standardzie DOM (zwanym też notacją DOM) zamiast w notacji CSS.

Oto konkretny przykład ilustrujący tę różnicę. Aby ustawić szerokość obramowania sekcji w notacji CSS, napisałbyś coś takiego:

```
div {  
border-style:solid;  
border-width:5px;  
}
```

Załóżmy teraz, że chciałbyś animować tę szerokość obramowania. W jQuery właściwość szerokości krawędzi ustawia się za pomocą notacji DOM, tak jak tutaj:

```
$("#div").animate({borderWidth:30  
},"slow");
```

Zauważ, że w notacji CSS dla właściwości szerokości piszesz `border-width`, natomiast w notacji DOM dla tej samej właściwości używasz zapisu `borderWidth`.

Jeśli chcesz poczytać więcej o różnicach między tymi dwoma notacjami, zapoznaj się z poniższym artykułem:

[http://www.oxfordu.net/webdesign/dom/straight\\_text.html](http://www.oxfordu.net/webdesign/dom/straight_text.html)

**P:** A co powinienem zrobić, jeśli chciałbym animować zmianę koloru?

**U:** Aby animować przejścia między kolorami, musisz skorzystać z jQuery UI, które zawiera więcej efektów, niż zostało dołączonych do jQuery. jQuery UI opiszemy w rozdziale 10., ale z pominięciem efektów. Kiedy już dowiesz się, jak pobierać i ustawiać motywy oraz dołączać jQuery UI do swoich aplikacji internetowych, animowanie zmian koloru stanie się bardzo proste.

## Dodaj funkcje animujące do swojego skryptu

Korzystając z kodu, który ułożyłeś w ćwiczeniu na poprzedniej stronie, zaktualizuj swój plik ze skryptem dla aplikacji Zrób sobie potwora.



```

$("#head").click(function(){
  if (headclix < 9){
    $(this).animate({left:"-=367px"},500);
    headclix+=1;
  }
  else{
    $(this).animate({left:"0px"},500);
    headclix = 0;
  }
});

```

*Możemy tu skorzystać ze słowa kluczowego „this”, ponieważ znajdujemy się wewnątrz funkcji obsługującej element, który kliknęliśmy.*

```

$("#eyes").click(function(){
  if (eyeclix < 9){
    $(this).animate({left:"-=367px"},500);
    eyeclix+=1;
  }
  else{
    $(this).animate({left:"0px"},500);
    eyeclix = 0;
  }
});

```

```

$("#nose").click(function(){
  if (noseclix < 9){
    $(this).animate({left:"-=367px"},500);
    noseclix+=1;
  }
  else{
    $(this).animate({left:"0px"},500);
    noseclix = 0;
  }
});

```

```

$("#mouth").click(function(){
  if (mouthclix < 9){
    $(this).animate({left:"-=367px"},500);
    mouthclix+=1;
  }
  else{
    $(this).animate({left:"0px"},500);
    mouthclix = 0;
  }
});

```



my\_scripts.js



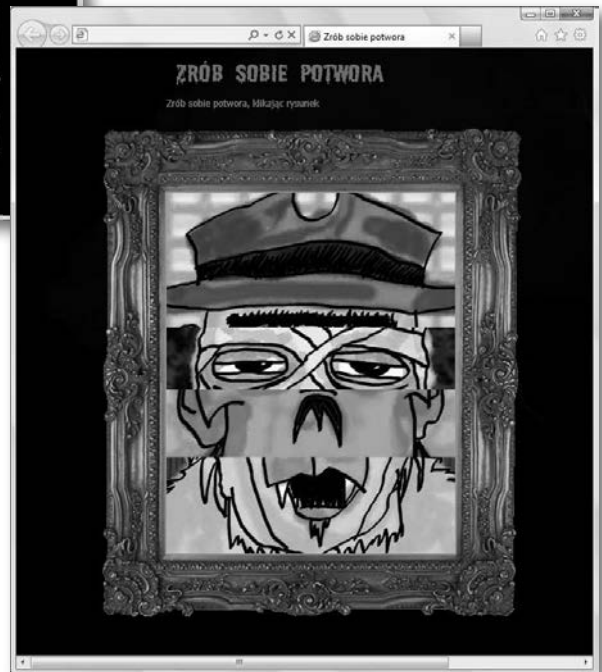
# Jazda próbna

Otwórz stronę w swojej ulubionej przeglądarce, aby upewnić się, że wszystko działa.



Udało Ci się  
napisać działający  
niestandardowy  
efekt przesuwania  
w lewo.

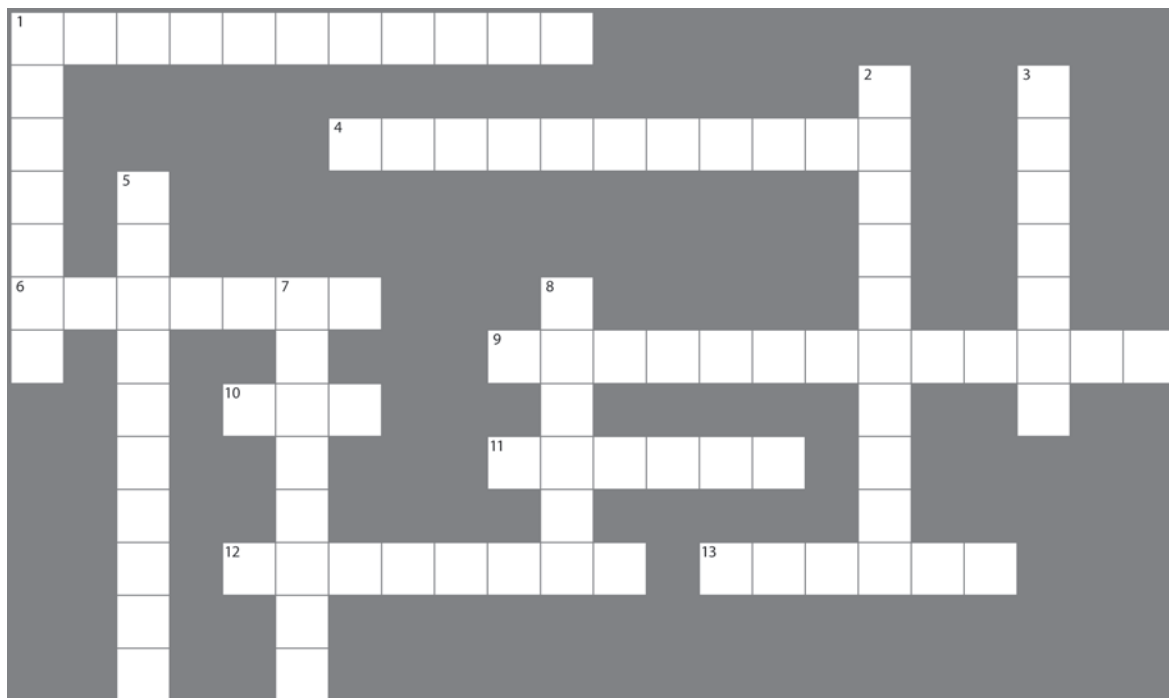
Za pomocą kilku kliknięć  
użytkownik może zrobić sobie  
własną twarz potwora.





# Krzyżówka jQuery

Nadszedł czas, aby wygodnie usiąść i dać swojej lewej półkuli jakieś zajęcie. Będzie to zwykła krzyżówka. Wszystkie wyrazy i wyrażenia stanowiące hasła pochodzą z bieżącego rozdziału.



## Poziomo

1. Metoda powodująca następujące działanie: jeżeli wskazany element ma wysokość 0, interpreter JS wysuwa go w dół, jeśli natomiast wskazany element ma pełną wysokość, interpreter JS wsuwa go do góry.
4. Tworzą iluzję elementu przesuwanego się po ekranie.
6. `hide`, `show` i `toggle` animują tę właściwość CSS.
9. Funkcjonalność jQuery umożliwiająca łączenie ze sobą metod, które chcesz wywołać dla danego zbioru elementów.
10. Efekty jQuery polegają na manipulowaniu nim w locie.
11. Tę właściwość CSS oraz właściwość `width` możesz animować w celu stworzenia iluzji rosnącego albo malejącego elementu.
12. Parametr kontrolujący czas trwania efektu.
13. Kiedy wywołasz ten efekt jQuery, interpreter JS zmienia przezroczystość wskazanego elementu od 0 do 100.

## Pionowo

1. 1000 milisekund.
2. Metoda `animate` będzie działać tylko dla właściwości CSS, które mają takie wartości.
3. Biblioteka jQuery oferuje tę metodę, jeśli zechcesz stworzyć efekty niestandardowe.
5. Efekt uzyskany za pomocą animowania właściwości `height` elementu.
7. Jedną z wartości, obok `fixed` i `relative`, jaką można nadać właściwości określającej położenie elementu w celu jego animowania.
8. Metoda nadająca danemu elementowi w animowany sposób określony stopień przezroczystości.

## Patrz, mam! Bez Flasha!

Dyrektor ds. projektów internetowych jest zadowolona z wyników Twojej pracy nad aplikacją Zrób sobie potwora. Skorzystałaś z gotowych efektów jQuery w połączeniu z własnymi niestandardowymi efektami skrojonymi na miarę potrzeb klienta.

# GRYZMOŁKI

Nasza docelowa grupa w wieku od 6 do 10 lat uwielbia stronę Zrób sobie potwora. I nie potrzebujemy używać Flasha ani wtyczek do przeglądarek. No proszę, jQuery idealnie wpasował się w nasze potrzeby.

Fajowo! Zrobiłem tyle potworów, że aż się w tym pogubiłem!

Niesamowite! Wystraszę moją młodszą siostrę potworem, którego zrobiłam!





# Rozwiązanie krzyżówki jquery

1	S	L	I	D	E	T	O	G	G	L	E													
E																2	N						3	A
K									4	E	F	E	K	T	Y	R	U	C	H	U				N
U		5	W																	M			I	
N			Y																	E			M	
6	D	I	S	P	L	7	A	Y					8	F						R			A	
A		U				B				9	Ł	A	Ń	C	U	C	H	Y	M	E	T	O	D	
		N			10	C	S	S							D					C			E	
		I				O						11	H	E	I	G	H	T		Z				
		Ę				L								T						N				
		C			12	D	U	R	A	T	I	O	N			13	F	A	D	E	I	N		
		I				T																		
		E				E																		



## Twój niezbędnik jQuery

Masz już za sobą rozdział 5., a do swojego niezbędnika dodałeś efekty wygaszania kolorów oraz wysuwania, a także niestandardowe animacje.

### Efekty wygaszania

Zmieniają właściwość CSS opacity wskazanego elementu:

fadeIn  
fadeOut  
fadeTo

### Efekty wysuwania

Zmieniają właściwość CSS height wskazanego elementu:

slideUp  
slideDown  
slideToggle

### animate

Umożliwia tworzenie niestandardowych animacji, kiedy gotowe efekty jQuery już nie wystarczają.

Animuje właściwości CSS.

Działa tylko z takimi właściwościami CSS, które przyjmują wartości liczbowe.

Elementy można przesuwac względnie albo bezwzględnie.

Połączenia operatorów (=, +, -) w znacznym stopniu ułatwiają animację względną.



# Skorowidz

## A

`addClass`, *Patrz:* metoda `addClass`  
`after`, *Patrz:* metoda `after`  
Ajax, 332, 334, 348, 356, 360, 368, 456  
`ajax`, *Patrz:* metoda `ajax`  
akcesor, 312  
alarm, 78, 86, 94  
`alert`, *Patrz:* alarm  
animacja, 211, 214, 218, 223, 228, 232, 235, 236, 238, 242, 244, 250, 292, 490, 492  
bezwzględna, 242  
kolejki, 490  
względna, 242  
`animate`, *Patrz:* metoda `animate`  
API, 449, 450, 451, 454, 460, 473, 481  
`append`, *Patrz:* metoda `append`  
`appendTo`, *Patrz:* metoda `appendTo`  
ASP, 347  
asynchroniczność, 334

## B

baza danych, 367  
MySQL, 371, 456  
serwer, 371  
`before`, *Patrz:* metoda `before`  
biblioteka jQuery, *Patrz:* jQuery biblioteka  
JavaScript, 41  
UI Effects, 492  
`button`, *Patrz:* metoda `button`

## C

CDN, 487  
`children`, *Patrz:* metoda `children`  
Chrome, *Patrz:* przeglądarka Google Chrome  
`clearInterval`, *Patrz:* metoda `clearInterval`  
`click`, *Patrz:* metoda `click`  
Cold Fusion, 347  
`contains`, *Patrz:* metoda `contains`  
content delivery networks, *Patrz:* CDN  
content distribution networks, *Patrz:* CDN  
Crockford Douglas, 391  
cross-site scripting, *Patrz:* skrypt między wiotrynami  
CSS, 30, 39, 40, 84, 88, 410  
`display`, 224  
identyfikator, 49, 50, 52, 59, 61, 77, 83, 86, 103  
klasa, 49, 50, 83, 84, 85, 86, 88, 89, 96, 97, 103, 151, 154, 166, 286, 292, 492  
opacity, 224, 225  
pozycjonowanie, 218

## D

dane  
baza, *Patrz:* baza danych  
sanityzacja, 368

serializacja, 368  
walidacja, 368, 442, 491  
`data`, *Patrz:* metoda `data`  
`datepicker`, *Patrz:* metoda `datepicker`  
`delay`, *Patrz:* metoda `delay`  
`dequeue`, *Patrz:* metoda `dequeue`  
`detach`, *Patrz:* metoda `detach`  
DOM, *Patrz:* HTML Obiektowy Model Dokumentu  
domknięcie, *Patrz:* JavaScript domknięcie  
Dragonfly, *Patrz:* przeglądarka Opera  
drive-by scripting, *Patrz:* skrypt przekierowujący  
Dump, 489

## E

`each`, *Patrz:* metoda `each`  
easing, 492  
EditPlus, 31  
efekty, *Patrz:* jQuery efekty  
`element`, 40, 42, 44, 45, 47, 49, 50, 51, 53, 59, 69, 71, 75, 78, 84, 85, 86, 87, 96, 102, 110, 115, 117, 121, 126, 127, 143, 164, 166, 185, 188, 204, 342, 460  
`empty`, *Patrz:* metoda `empty`  
`eq`, *Patrz:* metoda `eq`  
`equal`, *Patrz:* metoda `equal`  
eXtensible Markup Language, *Patrz:* XML

## F

`fadeIn`, *Patrz:* metoda `fadeIn`  
`fadeOut`, *Patrz:* metoda `fadeOut`  
`fadeOutTo`, *Patrz:* metoda `fadeOutTo`  
`filter`, *Patrz:* metoda `filter`  
`find`, *Patrz:* metoda `find`  
`first`, *Patrz:* metoda `first`  
formularz, 364, 409, 443  
pole tekstowe, 364  
lista rozwijana, 364  
walidacja, *Patrz:* dane walidacja  
funkcja, 48, 52, 69, 78, 115, 119, 136, 137, 148, 184, 254, 255, 256, 341, 494  
anonimowa, 137, 138  
`array_push`, 390  
change, 427  
create, 427  
czasowa, 230  
date, 351  
deklaracja, 138  
easingu, 412, 492  
`getJSON`, 405  
`getRandom`, 312, 315, *Patrz też:* wartość losowa  
`json_encode`, 390, 405  
losowa, 312, 315, *Patrz też:* wartość losowa  
losowa  
nazwa, 138

obsługi zdarzeń, 119, 292, 427  
reset, 319, 320  
samowolująca się, 347, 355, *Patrz też:* funkcja anonimowa  
slide, 427  
start, 427  
stop, 427  
wartość, 143, 150, 158  
zmiennie, 141, 142, 150, 158

## G

generator liczb losowych, 230  
`get`, *Patrz:* metoda `get`  
`getJSON`, *Patrz:* metoda `getJSON`  
`getScript`, *Patrz:* metoda `getScript`  
getter, 312  
Google  
powłoka, *Patrz:* powłoka przeglądarka, *Patrz:* przeglądarka Google Chrome  
Google Maps, 449, 450, 451, 453, 473  
geokodowanie, 473  
trasy dojazdów, 473

## H

`hide`, *Patrz:* metoda `hide`  
HTML, 30, 39, 40, 88, 334, 335, 350  
formularz, *Patrz:* formularz Obiektowy Model Dokumentu, 43, 44, 47, 115, 159, 176, 177, 185, 195, 210, 226, 333, 336  
sekcja, *Patrz:* sekcja  
Hypertext Processor, *Patrz:* PHP

## I

`inArray`, *Patrz:* metoda `inArray`  
indeks, 146, 262  
instrukcja  
else, 145, 158, 275, 389  
else if, 275, 280, 389  
if, 145, 275, 276  
insert, 374  
select, 381, 384  
warunkowa, 389  
interakcja, 412, 437  
Internet Explorer, *Patrz:* przeglądarka Internet Explorer

## J

JavaScript, 30, 41, 45, 47, 55, 88, 132, 333, 334  
domknięcie, 494, 495  
interpreter, 47, 115, 185, 224, 238  
logika warunkowa, *Patrz:* logika warunkowa  
operator, 243  
zmienna, 92

JavaScript Object Notation, 356

język

ASP, *Patrz:* ASP  
Cold Fusion, *Patrz:* Cold Fusion  
JSP, *Patrz:* JSP  
PHP, *Patrz:* PHP  
skryptowy, 40

jQuery

animacja, 211, 224  
biblioteka, 32, 71, 132, 251, 329, 412, 446, 492  
debugowanie kodu, 489  
efekty, 62, 211, 224, 234, 235  
funkcja, *Patrz:* funkcja  
szablon, 495  
tablica, *Patrz:* tablica  
UI, 244, 407, 412, 437, 446, 492  
wersja deweloperska, 32  
wersja produkcyjna, 32  
wtyczki, 489, 493  
zdarzenie, 78, 115, 118, 120, 127, 158, 348, 356

JSON, 260, 256, 361, 386, 387, 391, 397, 405, 456

JSP, 347

## K

kaskadowe arkusze stylów,

*Patrz:* CSS

Keyhole Markup Language,

*Patrz:* KML

KML, 336

kolejka, 490

komunikat alarmowy, 82

konkatenacja, 93

konstruktor, 256, 288, 452, 481

krzyżówka, 67, 247, 323, 403, 479

kwerenda, 51

## L

last, *Patrz:* metoda last

length, *Patrz:* metoda length

Lerdorf Rasmus, 391

licznik, 238, 265, 291, 292

lista

nienumerowana, 164  
rozwijana, 364, 419, 464

load, *Patrz:* metoda load

logika warunkowa, 144, 145, 158, 316, 356, 400

## Ł

łańcuch

metod, *Patrz:* metoda spinanie  
tekstowy, 86, 143, 185, 186, 253, 351, 368, 255, 386

## M

marker, 460

Math.floor, *Patrz:* metoda Math.floor

Math.random, *Patrz:* metoda Math.random

metoda, 71, 110, 254, 484, 485

addClass, 154

after, 195, 210

ajax, 340, 356, 360, 387

animate, 235, 236, 242, 250

append, 95

appendTo, 273

before, 195, 210

button, 422, 443, 446

children, 176, 177, 197, 198, 210, 342

clearInterval, 300, 326

click, 78, 81, 82

contains, 150

data, 490

datepicker, 417, 418, 446

delay, 299, 346

dequeue, 490

detach, 172, 184, 210

each, 204, 208, 342

empty, 184, 281, 282, 288

eq, 197, 210

equal, *Patrz:* metoda eq

fadeIn, 60, 63, 225, 232, 250

fadeOut, 225, 232, 250

fadeOut, 225, 250

filter, 198, 210

filtrująca, 197, 210

find, 208, 342

first, 197, 210

get, 356, 360

get, 367, 400

getJSON, 356, 360, 405

getScript, 356, 360

hide, 224, 232

inArray, 267, 268

last, 197, 210

length, 261

load, 356, 360

łańcuchowa, *Patrz:* metoda spinanie

Math.floor, 94, 107

Math.random, 94, 107

next, 176

noConflict, 488

not, 198, 210

onBlur, 292, 295, 326

onFocus, 292, 295, 326

parent, 176, 177, 178, 184, 210

parents, 184, 198

post, 356, 360, 367, 378, 387, 400

prev, 176

queue, 490

remove, 102, 107, 172, 184, 210

replaceWith, 190, 191, 193, 194, 210

serialize, 368

serializeArray, 368, 405

setInterval, 299, 301, 346, 356

setTimeout, 230, 232, 291, 299, 301, 326, 346,

show, 224

siblings, 184

slice, 198, 208, 210

slideDown, 62, 69, 226, 250

slideLeft, 227, 235

slideRight, 227, 235

slideToggle, 62, 69, 226, 250

slideUp, , 62, 69, 226, 250

spinanie, 178, 229

statyczna, 150

stop, 244

switch, 280

toggle, 224, 232

trigger, 127, 288

MySQL, 371, 380, 405

## N

next, *Patrz:* metoda next

noConflict, *Patrz:* metoda noConflict

not, *Patrz:* metoda not

## O

obiekt, 254, 257, 260, 261, 288, 386, 447, 451

API, 460

instancja, 256

jednokrotnego użycia, 260

JSON, *Patrz:* JSON

wielokrotnego użycia, 260

window, 292, 320, 326

właściwość, 255

XMLHttpRequest, 347

zakres ważności, 494

Obiektowa Notacja JavaScriptu,

*Patrz:* JSON

Obiektowy Model Dokumentu,

*Patrz:* HTML Obiektowy Model Dokumentu

onBlur, *Patrz:* metoda onBlur

onFocus, *Patrz:* metoda onFocus

OOXML, 336

Open Office XML, *Patrz:* OOXML

Opera, *Patrz:* przeglądarka Opera

operator, *Patrz:* JavaScript operator

kropka, 257, 386

logiczny, 276, 288, 389

porównania, 276, 288, 389

trójargumentowy, 276, 280

overlay, *Patrz:* powłoka

## P

para klucz/wartość, 389

parent, *Patrz:* metoda parent

parents, *Patrz:* metoda parents  
pętla

do...while, 265, 266, 288, 388  
for, 265, 266, 288, 388  
for...in, 273  
foreach, 388  
nieskończona, 265, 291  
while, 273, 276, 388  
PHP, 347, 356, 360, 367, 376, 383, 388, 389,  
391, 400, 405, 498, 502  
zagrożenia, 400  
PHPMyAdmin, 380  
post, *Patrz:* metoda post  
powłoka, 471  
prev, *Patrz:* metoda prev  
przeglądarka  
Dragonfly dla Opery, 30  
Firebug dla Firefox, 30  
Google Chrome, 30, 184, 301, 341  
Internet Explorer, 30, 301  
Mozilla Firefox, 301  
odświeżanie, 42  
Safari, 30  
silnik, 218, 238  
przycisk, 61, 102, 169, *Patrz też:* metoda  
button, widget przycisku  
PSPad, 31

## Q

queue, *Patrz:* metoda queue

## R

RDBMS, 371  
RDF Site Summary, *Patrz:* RSS  
Real Simple Syndication, *Patrz:* RSS  
Relational Database Management Systems,  
*Patrz:* RDBMS  
remove, *Patrz:* metoda remove  
replaceWith, *Patrz:* metoda  
replaceWith  
RSS, 336, 360

## S

Safari, *Patrz:* przeglądarka Safari  
Scalable Vector Graphics, *Patrz:* SVG  
sekcja, 75  
sekwencjonowanie akcji, 490  
selektor, 45, 48, 49, 69, 71, 83, 87, 99, 110, 119,  
166, 184, 342, 486  
CSS, 50, 59, 87  
identyfikujący, *Patrz:* CSS  
identyfikator  
potomka, 99, 110  
this, 99, 107, 110, 204, 208  
znacznika, *Patrz:* selektor

serializacja danych, *Patrz:* dane  
serializacja  
serialize, *Patrz:* metoda serialize  
serializeArray, *Patrz:* metoda serializeArray  
serwer

Apache, 31, 499, 502  
MySQL, 31, 350, 498, 500, 504  
PHP, 31, 350, 498, 499, 502  
WWW, 31, 498, 499, 501  
setInterval, *Patrz:* metoda setInterval  
setTimeout, *Patrz:* metoda setTimeout  
show, *Patrz:* metoda show  
siblings, *Patrz:* metoda siblings  
silnik selektorowy, 45  
Simple Object Access Protocol, *Patrz:* SOAP  
skrypt, 40, 61, 88, 128  
między witrynami, 400  
PHP, 405  
przekierowujący, 400  
slice, *Patrz:* metoda slice  
slideDown, *Patrz:* metoda slideDown  
slideLeft, *Patrz:* metoda slideLeft  
slideRight, *Patrz:* metoda slideRight  
slideToggle, *Patrz:* metoda slideToggle  
slideUp, *Patrz:* metoda slideUp  
słowo kluczowe  
as, 388  
echo, 389  
function, 494  
print, 389  
słuchacz zdarzeń, 115, 116, 121, 123, 474  
SOAP, 336, 360  
spinanie, *Patrz:* metoda spinanie  
SQL, 405  
stop, *Patrz:* metoda stop  
strona, 88  
interaktywna, 38, 57  
stacyczna, 38  
struktura, 44, 88  
suwak, 426, 442, 446  
SVG, 336, 360  
switch, *Patrz:* metoda switch  
Systemy Zarządzania Relacyjnymi Bazami  
Danych, *Patrz:* RDBMS  
szablon, *Patrz:* jQuery szablon

## T

tablica, 186, 187, 203, 204, 208, 210, 253, 255,  
261, 263, 288, 385, 386, 388, 490  
asocjacyjna, 367, 389, 390, 397  
terminator, 184  
TextPad, 31  
TextWrangler, 31  
this, 99, 110, 204, 208, 493  
toggle, *Patrz:* metoda toggle  
trigger, *Patrz:* metoda trigger

## U

UML, 255, 260  
Unified Modeling Language, *Patrz:* UML

## V

Variable-Debugger, 489

## W

walidacja  
danych, *Patrz:* dane walidacja  
formularzy, *Patrz:* dane walidacja  
wałek malarski, 437  
wartość losowa, 92, 147, 311, *Patrz też:*  
funkcja losowa  
wersja  
deweloperska, 32  
produkcyjna, 32  
widget, 412, 419, 446  
autouzupelniania, 442  
button, 446  
datepicker, 446  
o zmiennych rozmiarach, 437  
przeciągalny, 437  
przycisku, 422  
sortowalny, 437  
upuszczalny, 437  
wybieralne, 437  
wstrzyknięcie kodu SQL,  
wtyczka, 329, 410  
idTabs, 329  
w widgetami, 412  
z efektami, 412  
z interakcjami, 412

## X

XHTML, 336  
XML, 334, 335, 336, 350, 356, 360, 361, 363,  
391

## Z

zagrożenia, 400  
zasada tożsamego pochodzenia, 341  
zdarzenie, *Patrz:* jQuery zdarzenie  
hover, 153  
nasłuchiwanie, 474  
słuchacz, *Patrz:* słuchacz zdarzeń  
usuwanie, 122  
wiązanie, 117  
zmienna, 92, 93, 96, 184, 185, 253, 388  
lokalna, 494  
zakres ważności, 494  
Zunifikowany Język Modelowania, *Patrz:*  
UML  
zwracanie, 51, 87



# 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>

Władaj stronami z jQuery!

# jQuery. Rusz głową!

Jeżeli Twoja witryna internetowa ma się wyróżniać na tle konkurencji, musi być interaktywna, przyjemna dla oka i wygodna w użyciu. Jeżeli nie spełnia tych warunków, będzie jej niezwykle trudno zdobyć popularność. Ale zawsze możesz wykorzystać JavaScript! Zastosowanie tego języka programowania pozwala na istotne wzbogacenie strony. Z kolei użycie biblioteki jQuery sprawia, że wykorzystanie potencjału JavaScriptu jeszcze nigdy nie było tak proste. Jeżeli dodać do tego liczne gotowe do użycia efekty specjalne, otrzymamy niemalże perfekcyjne rozwiązanie.

Kolejna książka z serii *Rusz głową!* to najlepszy sposób na opanowanie możliwości biblioteki jQuery. Atrakcyjna forma graficzna oraz nowoczesna metodologia nauczania sprawiają, że przyswajanie wiedzy jest przyjemne i efektywne. Autorzy nie wymagają od Ciebie znajomości języka JavaScript — dzięki temu możesz z marszu zacząć poznawać jQuery. Nie ma na co czekać! Sprawdź, jak używać selektorów, reagować na zdarzenie, modyfikować drzewo DOM oraz używać efektów specjalnych. *jQuery. Rusz głową!* to Twoja przepustka do tworzenia angażujących, interaktywnych witryn WWW, które wyglądają i działają tak jak prawdziwe aplikacje!

▼  
**Wykorzystaj efekty specjalne**

▼  
**Bez trudu modyfikuj drzewo DOM**

▼  
**Nasłuchuj zdarzeń**

▼  
**Zaskocz swoich użytkowników**

**Poznaj jQuery i ulepsz swoją stronę internetową!**

**helion.pl**  
księgarnia internetowa

Nr katalogowy: 8634

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/nowosci>

**Helion SA**

ul. Kościuszki 1c, 44-100 Gliwice

tel.: 32 230 98 63

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

<http://helion.pl>

sięgnij po WIECEJ



KOD KORZYŚCI

ISBN 978-83-246-3864-2



Cena 77,00 zł

Informatyka w najlepszym wydaniu

9 788324 638642

