

Adriaan de Jonge  
Phil Dutson



# jQuery, jQuery UI oraz jQuery Mobile

## Receptury

Najlepsze receptury dla jQuery!

Tytuł oryginału: jQuery, jQuery UI, and jQuery Mobile: Recipes and Examples

Tłumaczenie: Piotr Rajca

ISBN: 978-83-246-7703-0

Authorized translation from the English language edition, entitled: JQUERY, JQUERY UI, and JQUERY Mobile: Recipes and Examples; ISBN 0321822080; by Adriaan De Jonge; and by Phillip Dutson; published by Pearson Education, Inc, publishing as Addison Wesley.  
Copyright © 2013 Pearson Education, Inc.

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

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

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)

Pliki z przykładami omawianymi w książce można znaleźć pod adresem:  
<ftp://ftp.helion.pl/przyklady/jquere.zip>

Drogi Czytelniku!  
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres  
<http://helion.pl/user/opinie/jquery>  
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

# Spis treści

<b>O autorach .....</b>	<b>15</b>
<b>Wstęp .....</b>	<b>17</b>
Dlaczego powstała książka z recepturami i przykładami na temat jQuery? .....	17
Dla kogo jest przeznaczona ta książka? .....	18
Jak korzystać z tej książki? .....	18
Struktura książki .....	19
Zasoby dodatkowe .....	19
<b>Część I Podstawy .....</b>	<b>21</b>
<b>Rozdział 1. Rozpoczynanie pracy z jQuery .....</b>	<b>23</b>
Receptura: Przedstawienie podstawowego sposobu użycia jQuery .....	23
Receptura: Stosowanie jQuery wraz z innymi bibliotekami .....	26
Receptura: Określanie wersji biblioteki jQuery .....	27
Receptura: Przeglądanie tablic przy użyciu funkcji each() .....	28
Receptura: Operacje na tablicach przy użyciu funkcji map() .....	30
Receptura: Operowanie na tablicach elementów .....	31
Receptura: Określanie położenia elementu przy użyciu funkcji index() .....	32
Receptura: Odnajdywanie elementów tablicy przy użyciu funkcji grep() .....	33
Receptura: Określanie wielkości zbioru elementów przy użyciu funkcji length() .....	34
Receptura: Pobieranie atrybutów data-HTML5 .....	35
Receptura: Zapis danych przy użyciu funkcji data() .....	36
Receptura: Usuwanie danych przy użyciu metody removeData() .....	38
Receptura: Testowanie zmiennych i operowanie na nich .....	39
Receptura: Rozszerzanie obiektów przy użyciu funkcji extend() .....	41
Receptura: Serializacja danych z formularzy .....	43
Receptura: Testowanie dostępnych możliwości przeglądarki .....	44
Podsumowanie .....	46

<b>Rozdział 2. Wybór elementów .....</b>	<b>47</b>
Receptura: Łączenie dwóch zbiorów elementów przy użyciu funkcji add() .....	47
Receptura: Precyzowanie zapytania przy użyciu funkcji filter() .....	48
Receptura: Wybieranie elementów podrzędnych przy użyciu funkcji find() oraz children() .....	49
Receptura: Wybór elementów za pomocą funkcji has() oraz sprawdzanie ich przy użyciu funkcji is() .....	51
Receptura: Wybieranie pól formularzy przy wykorzystaniu pseudoselektorów .....	52
Receptura: Zagnieżdżanie selektorów .....	53
Receptura: Emulacja selektora hover CSS .....	54
Receptura: Wybieranie tekstu za pomocą funkcji contains() .....	55
Przykład: Wyróżnianie pojedynczego słowa .....	56
Receptura: Tworzenie własnych selektorów .....	57
Receptura: Ograniczanie zakresu kontekstu wyboru .....	59
Podsumowanie .....	60
<b>Rozdział 3. Modyfikowanie stron .....</b>	<b>61</b>
Receptura: Dodawanie klas .....	61
Przykład: Usuwanie klas .....	62
Receptura: Generacja nazw klas .....	63
Receptura: Zmiana właściwości i atrybutów .....	65
Receptura: Zmiana kodu HTML wewnątrz elementu .....	66
Receptura: Dodawanie zawartości przy użyciu funkcji append() oraz appendTo() .....	67
Przykład: Dodawanie wyniku funkcji .....	68
Receptura: Dodawanie treści przy użyciu funkcji prepend() oraz prependTo() .....	70
Receptura: Dynamiczna generacja kodu HTML przy wykorzystaniu jQuery .....	71
Receptura: Dołączanie i odłączanie elementów .....	72
Receptura: Kopiowanie elementów przy użyciu funkcji clone() .....	73
Receptura: Wstawianie elementów w określonym miejscu .....	75
Przykład: Przesuwanie elementów w górę i w dół listy .....	76
Receptura: Usuwanie elementów .....	78
Receptura: Zagnieżdżanie elementów i usuwanie elementów otaczających .....	80
Podsumowanie .....	83
<b>Rozdział 4. Odbieranie zdarzeń i odpowiadanie na nie .....</b>	<b>85</b>
Receptura: Obsługa zdarzeń związanych z myszą .....	85
Przykład: Rysowanie na elemencie canvas .....	87
Receptura: Obsługa zdarzeń klawiatury .....	89
Receptura: Obsługa zdarzeń formularzy .....	90

Receptura: Obsługa zdarzeń związanych z przewijaniem .....	92
Receptura: Dodawanie głównych procedur obsługi zdarzeń przy użyciu funkcji live() oraz die() .....	93
Receptura: Przekazywanie zdarzeń do konkretnego elementu nadrzędnego przy użyciu funkcji delegate() .....	95
Receptura: Zmiana kontekstu wykonania funkcji przy użyciu funkcji proxy() .....	98
Podsumowanie .....	99
<b>Rozdział 5. Komunikacja z serwerem .....</b>	<b>101</b>
Receptura: Przygotowanie testowego serwera z użyciem Node.js .....	101
Receptura: Generacja żądania GET .....	103
Receptura: Bezpośrednie wczytywanie kodu HTML .....	104
Receptura: Obsługa żądań przy użyciu obietnic .....	105
Receptura: Obsługa błędów serwera .....	107
Receptura: Obsługa problemów z odnalezieniem strony .....	108
Receptura: Obsługa przekierowań .....	110
Receptura: Określanie czasu oczekiwania na odpowiedź .....	111
Receptura: Przekazywanie nagłówków HTTP .....	112
Przykład: Weryfikacja danych z formularza po stronie serwera .....	113
Receptura: Wczytywanie kodu XML .....	115
Receptura: Obsługa zdarzeń AJAX .....	116
Receptura: Czytanie danych JSONP z zewnętrznego serwera .....	117
Podsumowanie .....	119
<b>Część II Interfejs użytkownika .....</b>	<b>121</b>
<b>Rozdział 6. Interakcja z użytkownikiem .....</b>	<b>123</b>
Pobieranie i instalacja jQuery UI .....	123
Receptura: Przeciąganie elementów .....	124
Określanie postaci komponentów draggable .....	125
Określanie opcji komponentów draggable .....	126
Obsługa zdarzeń komponentów draggable .....	129
Wywoływanie metod komponentów draggable .....	130
Receptura: Upuszczanie elementów .....	130
Określanie stylu komponentów droppable .....	132
Określanie opcji komponentów droppable .....	132
Przechwytywanie zdarzeń komponentów droppable .....	132
Wywoływanie metod komponentów droppable .....	134
Receptura: Zmiana kolejności elementów przy wykorzystaniu elementów sortowalnych .....	135
Określanie stylów elementów sortowalnych .....	136
Określanie opcji elementów sortowalnych .....	136
Obsługa zdarzeń elementów sortowalnych .....	136
Wywoływanie metod elementów sortowalnych .....	140

Przykład: Sortowanie elementów tworzących strukturę drzewiastą ...	141
Receptura: Zaznaczanie elementów listy wypunktowanej .....	142
Style komponentów selectable .....	144
Określanie opcji komponentów selectable .....	144
Obsługa zdarzeń komponentów selectable .....	144
Wywoływanie metod komponentów selectable .....	146
Przykład: Zaznaczanie elementów w strukturze drzewiastej .....	146
Receptura: Zmiana wielkości elementów .....	148
Określanie wyglądu elementów o zmiennej wielkości .....	149
Określanie opcji elementów o zmiennej wielkości .....	149
Obsługa zdarzeń elementów o zmiennej wielkości .....	150
Wywoływanie metod elementów o zmiennej wielkości .....	151
Podsumowanie .....	152
<b>Rozdział 7. Interakcja z widżetami .....</b>	<b>153</b>
Receptura: Grupowanie treści przy użyciu akordeonu .....	153
Określanie wyglądu akordeonu .....	154
Określanie opcji akordeonu .....	155
Obsługa zdarzeń akordeonu .....	157
Metody .....	158
Receptura: Sugerowanie wartości przy użyciu widżetu automatycznego uzupełniania .....	159
Określanie wyglądu widżetu automatycznego uzupełniania .....	161
Określanie opcji widżetu automatycznego uzupełniania .....	161
Obsługa zdarzeń widżetu automatycznego uzupełniania .....	161
Wywoływanie metod widżetu automatycznego uzupełniania .....	164
Receptura: Przekształcanie elementów w przyciski .....	165
Określanie wyglądu przycisków .....	166
Określanie opcji przycisków .....	167
Obsługa zdarzeń przycisków .....	167
Metody .....	168
Przykład: Określanie wyglądu przycisków opcji przy użyciu widżetu buttonset .....	168
Receptura: Wybór dat przy użyciu widżetu datepicker .....	169
Określanie wyglądu kalendarza .....	170
Określanie opcji kalendarza .....	171
Obsługa zdarzeń kalendarza .....	176
Wywoływanie metod kalendarza .....	177
Receptura: Przyciąganie uwagi przy użyciu okien dialogowych .....	178
Określanie wyglądu okna dialogowego .....	179
Określanie opcji okna dialogowego .....	180
Obsługa zdarzeń generowanych przez okna dialogowe .....	181
Wywoływanie metod okien dialogowych .....	184

Receptura: Wyświetlanie informacji o postępie prac przy wykorzystaniu widżetu progressbar .....	185
Określanie wyglądu paska postępów .....	186
Ustawianie opcji paska postępów .....	186
Obsługa zdarzeń paska postępów .....	186
Wywoływanie metod paska postępów .....	187
Receptura: Pobieranie liczb przy wykorzystaniu suwaka .....	187
Określanie wyglądu suwaka .....	188
Określanie opcji suwaków .....	189
Obsługa zdarzeń suwaka .....	189
Wywoływanie metod suwaków .....	190
Receptura: Nawigowanie po stronie przy wykorzystaniu kart .....	191
Określanie wyglądu kart .....	192
Określanie opcji widżetu kart .....	193
Obsługa zdarzeń widżetu tabs .....	194
Wywoływanie metod widżetu tabs .....	196
Podsumowanie .....	197

### **Część III Strony na urządzenia przenośne .....199**

#### **Rozdział 8. Zmiany wyglądu .....201**

Zmiana stylów używanych w komponentach jQuery UI .....	201
Tworzenie animacji z wykorzystaniem możliwości biblioteki jQuery Core .....	202
Receptura: Animacja kolorów przy użyciu jQuery UI .....	204
Receptura: Ukrywanie elementów przy użyciu funkcji wygaszania i przesuwania jQuery Core .....	205
Receptura: Dodawanie efektów graficznych przy użyciu jQuery UI .....	206
Receptura: Animacja dodawania i usuwania klas CSS przy użyciu jQuery UI .....	208
Receptura: Wyświetlanie wszystkich funkcji przejścia definiowanych przez jQuery UI .....	210
Receptura: Wyświetlanie wszystkich ikon definiowanych przez jQuery UI .....	211
Receptura: Użycie obietnic do wykonania kodu po zakończeniu animacji .....	213
Receptura: Wykonywanie kodu podczas animacji przy wykorzystaniu funkcji queue i dequeue .....	215
Podsumowanie .....	217

#### **Rozdział 9. Nawigacja przy użyciu jQuery Mobile .....219**

Receptura: Przygotowywanie podstawowych elementów jQuery Mobile .....	219
Receptura: Udostępnianie wielu stron przy użyciu jednego pliku HTML .....	221

Receptura: Zmiana tytułu elementu .....	223
Receptura: Wczytywanie zewnętrznych stron przy użyciu technologii AJAX .....	224
Receptura: Wyświetlanie komunikatów o wczytywaniu stron .....	227
Receptura: Odwołania do zewnętrznych stron bez korzystania z technologii AJAX .....	228
Receptura: Określanie animacji przejścia .....	229
Receptura: Stosowanie niestandardowych efektów przejść .....	231
Receptura: Obsługa zdarzeń generowanych przez urządzenia przenośne .....	233
Receptura: Generacja przycisku Wstecz .....	237
Receptura: Dodawanie stopek u dołu strony .....	238
Receptura: Ta sama stopka na wielu stronach .....	241
Receptura: Wyświetlanie stopki w ustalonym miejscu .....	242
Receptura: Wyświetlanie i ukrywanie stopki .....	244
Receptura: Optymalizacja nagłówków i stopek pod kątem prezentacji pełnoekranowych .....	246
Receptura: Zmiana schematu kolorów przy użyciu tematów .....	247
Receptura: Tworzenie wielu kolumn .....	249
Receptura: Zmiana stron przy wykorzystaniu skryptów .....	251
Receptura: Wczytywanie stron przy użyciu skryptów .....	253
Receptura: Dołączanie danych do węzłów DOM przy użyciu jQuery Mobile .....	255
Receptura: Korzystanie z funkcji pomocniczych jQuery Mobile .....	256
Podsumowanie .....	260
<b>Rozdział 10. Obsługa interakcji przy użyciu jQuery Mobile .....</b>	<b>261</b>
Receptura: Wyświetlanie pasków narzędzi w treści strony .....	261
Receptura: Wyświetlanie paska ostrzeżenia .....	263
Receptura: Dodawanie pasków menu do stopek .....	264
Receptura: Poruszanie się po stronach przy użyciu paska nawigacyjnego .....	265
Receptura: Wyświetlanie i ukrywanie treści przy użyciu elementów zwijanych .....	268
Receptura: Symulacja akordeonu poprzez użycie zbioru elementów zwijanych .....	270
Receptura: Pobieranie prostych danych tekstowych przy wykorzystaniu pól formularzy .....	271
Receptura: Wprowadzanie dat .....	273
Receptura: Wyświetlanie pól formularzy przy wykorzystaniu alternatywnych klawiatur .....	275
Receptura: Wyświetlanie wyspecjalizowanych rodzajów pól formularzy .....	276
Receptura: Wprowadzanie liczb całkowitych przy użyciu suwaków ....	278
Receptura: Ustawianie wartości binarnych przy użyciu przełączników .....	279



Receptura: Wybór jednego elementu przy użyciu przycisków opcji .....	280
Receptura: Wybieranie wielu elementów przy użyciu pól wyboru .....	282
Receptura: Wybieranie elementów z rozwijanej listy .....	284
Receptura: Wyświetlanie standardowych pól przez wyłączenie jQuery Mobile .....	286
Receptura: Wyświetlanie list elementów .....	288
Receptura: Filtrowanie elementów list .....	296
Receptura: Grupowanie elementów formularzy na listach .....	297
Podsumowanie .....	298

## **Część IV Wtyczki .....299**

### **Rozdział 11. Tworzenie wtyczek .....301**

Receptura: Przygotowywanie prostej, statycznej wtyczki .....	301
Receptura: Tworzenie prostych wtyczek kontekstowych .....	303
Receptura: Łączenie wtyczek i funkcji jQuery w łańcuch wywołań .....	304
Receptura: Parametryzacja wtyczek .....	306
Receptura: Tworzenie parametrów domyślnych .....	307
Receptura: Sterowanie działaniem wtyczki przy użyciu metod .....	308
Receptura: Tworzenie wtyczki, która tworzy wtyczki .....	311
Receptura: Rejestracja i wywoływanie funkcji zwrotnych .....	314
Receptura: Przekazywanie kontekstu do funkcji zwrotnych .....	316
Receptura: Zwracanie obiektu Deferred w celu wywołania odrębnych funkcji zwrotnych w razie powodzenia i porażki .....	317
Receptura: Zwracanie obiektu Promise w celu ukrycia szczegółów działania .....	319
Receptura: Przedstawienie zabezpieczenia obiektu obietnicy .....	320
Receptura: Stosowanie obiektów Promise do kontroli sterowania realizacją kodu .....	321
Receptura: Wizualizacja postępów przed wywołaniem końcowej funkcji zwrotnej .....	323
Receptura: Przekazywanie kontekstu do funkcji zwrotnych .....	324
Receptura: Przekazywanie kontekstu do funkcji informujących o postępach operacji .....	326
Podsumowanie .....	327

### **Rozdział 12. Korzystanie z gotowych wtyczek .....329**

Receptura: Wyświetlanie okna modalnego .....	330
Receptura: Stosowanie rozwijanego menu .....	332
Receptura: Stosowanie wtyczki ScrollSpy .....	334
Receptura: Przełączane karty .....	336
Receptura: Dodawanie podpowiedzi .....	338
Receptura: Stosowanie okienek informacyjnych .....	340
Receptura: Ostrzeganie użytkownika .....	342
Receptura: Przyciski .....	343

14 jQuery, jQuery UI oraz jQuery Mobile. Receptury

Receptura: Zwijanie treści .....	346
Receptura: Umieszczanie treści w karuzeli .....	348
Receptura: Korzystanie z automatycznego uzupełniania .....	351
<b>Skorowidz .....</b>	<b>355</b>

# Nawigacja przy użyciu jQuery Mobile

Ten rozdział zawiera podstawowe informacje na temat biblioteki jQuery Mobile. Opisuje on platformę służącą do przygotowywania stron, przechodzenia pomiędzy stronami, wczytywania zawartości przy użyciu technologii AJAX oraz tworzenia płynnych animacji przejść. W ramach samych stron można wyróżnić podstawowe elementy, takie jak nagłówki i stopki, które mogą zachowywać się różnie, w zależności od wybranych opcji konfiguracyjnych. Te podstawowe ustawienia strony, w połączeniu z takimi podstawowymi elementami jak kolumny, zdarzenia oraz prosty kod JavaScript, stanowią właśnie podstawy stosowania biblioteki jQuery Mobile.

## Receptura: Przygotowywanie podstawowych elementów jQuery Mobile

jQuery Mobile to odrębna biblioteka, niezależna do jQuery Core oraz jQuery UI. Przeważającej większości jej możliwości można używać bez konieczności pisania choćby jednego wiersza kodu JavaScript. Zamiast tego dodaje się do elementów HTML odpowiednie atrybuty. Wszystkie te atrybuty rozpoczynają się od prefiksu `data-`, podobnie jak niestandardowe atrybuty przedstawione w rozdziale 1., „Rozpoczynanie pracy z jQuery”, przy okazji prezentacji funkcji `data()`. Biblioteka jQuery Mobile używa w tym celu innej funkcji: `jqmData()`. Zasady korzystania z niej zostały przedstawione w dalszej części tego rozdziału.

jQuery Mobile wymaga, by tworzone strony WWW były przygotowywane w specyficznym sposób. Listing 9.1 przedstawia podstawową strukturę strony korzystającej z biblioteki jQuery Mobile, posiadającej nagłówek oraz bardzo prostą zawartość.

## Listing 9.1. Wyświetlanie prostej strony z nagłówkiem

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>jQuery Mobile: podstawy</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14   </head>
15   <body>
16
17   <div data-role="page">
18
19     <div data-role="header">
20       <h1>Tytuł strony</h1>
21     </div>
22
23     <div data-role="content">
24       <p>Witaj, świecie!</p>
25     </div>
26
27   </div>
28
29 </body>
30 </html>

```

W powyższym kodzie HTML można zwrócić uwagę na kilka rzeczy. Po pierwsze, w wierszach 4. i 5. jest określany metaznacznik `viewport`. Znacznik ten prosi urządzenie o ustawienie odpowiedniego poziomu powiększenia oraz wielkości strony i dostosowanie ich do aktualnie wyświetlanej zawartości. Ma to ogromnie duże znaczenie podczas tworzenia stron przeznaczonych dla urządzeń mobilnych. Domyślna wartość tego metaznacznika zależy od przeglądarki, lecz zazwyczaj wynosi około 980 pikseli. Jeśli rozdzielczość urządzenia jest mniejsza lub większa od domyślnej, to układ strony może zostać zaburzony. W takich przypadkach strona może się wydawać zbyt mała lub poziomy powiększenia zostanie ustawiony w taki sposób, że tekst będzie zbyt mały do wygodnego czytania i trzeba go będzie powiększyć, aby móc korzystać ze strony. Jednak dzięki określeniu szerokości strony (`width`) oraz jej początkowej skali (`initial-scale`) można dostosować wielkość zawartości do rozmiarów ekranu urządzenia.

W wierszach 6. i 7. jest określany używany arkusz stylów CSS. Przy jego użyciu można zmieniać kolory. Jeśli ktoś jest naprawdę odważny, to może go użyć do całkowitej zmiany wyglądu strony. W takich przypadkach należy jednak zastanowić się na liczbie urządzeń, na jakich strona zostanie przetestowana. Domyślnie działanie biblioteki jQuery Mobile jest testowane na bardzo dużej liczbie urządzeń.

W wierszach 8. – 13. znajdują się znaczniki pobierające zarówno bibliotekę jQuery Core, jak i jQuery Mobile. W kolejnych przykładach przedstawionych w dalszej części książki, ze względu na wydajność działania, kod JavaScript jest umieszczany na końcu strony. W przypadku korzystania z biblioteki jQuery Mobile zaleca się, by kod HTML był wczytywany na początku. Odnośniki tworzone przy użyciu jQuery Mobile wczytują zawartość stron docelowych za pomocą żądań asynchronicznych wykonywanych przy użyciu technologii AJAX (będzie o tym mowa w dalszej części rozdziału). Po wczytaniu takiej strony skrypty umieszczone w jej nagłówku zostaną zignorowane. Wszystkie skrypty, które mają zostać wykonane, gdyż są ważne dla danej strony, muszą zostać umieszczone w jej ciele.

Sama zawartość strony została umieszczona w wierszach 17. – 27. W przeważającej części jest to zwyczajny kod HTML. Jednak każdy użyty element `div` zawiera atrybut `data-role`, zawierający wartość: `page`, `header` lub `content`.

Znaczniki te oraz określane przez nie role są używane przez jQuery Mobile do przypisania poszczególnym elementom strony odpowiednich stylów, tematów i działania. Wyniki można obejrzeć po wyświetleniu tej strony w przeglądarce — a najlepiej przeglądarce mobilnej.

## Receptura: Udostępnianie wielu stron przy użyciu jednego pliku HTML

W jednym pliku HTML można umieścić wiele stron. Nie ma wtedy potrzeby przesyłania żądań na serwer, dzięki czemu interfejs aplikacji może reagować na poczynania użytkownika bardzo szybko i płynnie. Co więcej, aplikacja zachowuje zdolność przechodzenia pomiędzy poszczególnymi stronami, nawet gdy urządzenie nie będzie podłączone do internetu. Listing 9.2 pokazuje, w jaki sposób można stworzyć odnośnik do drugiej strony.

Takie rozwiązanie spisuje się bardzo dobrze w przypadkach, gdy aplikacja składa się z niewielkiej liczby stron. Jednak w pewnym momencie, zależnym od grupy urządzeń docelowych, problemem stanie się pojemność pamięci. Tworzenie odnośników do stron zewnętrznych zostało opisane w dalszej części rozdziału.

Listing 9.2. Przechodzenie na inną stronę

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Strony</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10   </script>
11   <script type="text/javascript" src=
```

```

12     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13   </script>
14 </head>
15 <body>
16
17 <div data-role="page">
18   <div data-role="header">
19     <h1>Pierwsza</h1>
20   </div>
21
22   <div data-role="content">
23     <p>Witaj, świecie <a href="#second">i przejdź na inną stronę.</a></p>
24   </div>
25
26   <!--
27   Bądź też... jeśli treść docelowa jest określana przy
28   użyciu elementu LINK, a nie umieszczona w elemencie DIV
29
30   <div data-role="content">
31     <p>Lub... <a href="#second" data-rel="dialog"
32       data-transition="pop">pokaż tę samą stronę jako
33       okno dialogowe!</a></p>
34   </div>
35   -->
36 </div>
37
38 <div data-role="page" id="second">
39
40   <div data-role="header">
41     <h1>Druga</h1>
42   </div>
43   <div data-role="content">
44     Żegnaj, świecie!
45   </div>
46
47 </div>
48
49 </body>
50 </html>

```

Zacznijmy od końca strony. W wierszach 38. – 45. został umieszczony element `div` określający zawartość drugiej strony. Ta strona ma swój własny nagłówek i swoją własną zawartość. Co ważniejsze, jej element `div` zawiera atrybut `id`.

W wierszach 22. – 24. została umieszczona zawartość pierwszej strony. W jej skład wchodzi także odnośnik do drugiej strony. Atrybut `id` służy jako kotwica, do której można się odwołać, umieszczając przed jej nazwą znak `#`. Warto zwrócić uwagę na wygląd adresu URL wyświetlanego na pasku adresu przeglądarki, po kliknięciu odnośnika do drugiej strony. Ten adres URL można zapisać na liście zakładek, by później przejść bezpośrednio do drugiej strony.

W końcu, wiersze 26. – 35. przedstawiają alternatywną wersję opisanej wcześniej zawartości strony. Aktualnie nie jest ona używana, gdyż została umieszczona pomiędzy znacznikami `<!--` oraz `-->`. Jeśli opisane wcześniej wiersze 22. – 24. zostaną zastąpione tą

zawartością, to druga strona zostanie wyświetlona jako okno dialogowe, a nie nowa strona. Jak można zobaczyć, to element odnośnika określa, czy strona docelowa zostanie wyświetlona jako normalna strona, czy też jako okno dialogowe.

Zaleca się, by nie mieszać sposobów wyświetlania pojedynczej strony. Należy ją wyświetlać jako stronę lub jako okno dialogowe. Zmiana sposobu wyświetlania stron w ramach jednego dokumentu HTML może prowadzić do nieoczekiwanego zachowania aplikacji.

## Receptura: Zmiana tytułu elementu

Jeden dokument HTML zawiera tylko jeden element `title`. Umieszczając wiele stron w jednym dokumencie HTML, można doprowadzić do ich nieodpowiedniego działania. Może się okazać, że strony umieszczone w jednym dokumencie HTML będą musiały zostać rozdzielone i zapisane w odrębnych dokumentach.

Listing 9.3 pokazuje, w jaki sposób można określać tytuły poszczególnych stron umieszczonych w tym samym dokumencie HTML.

Listing 9.3. Określanie odrębnych tytułów dla poszczególnych stron

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Tytuł strony</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14   </head>
15   <body>
16
17   <div data-role="page" id="first" data-title="Tytuł pierwszej strony">
18     <!-- Tytuł nie jest określany podczas wczytywania! Jeśli
19     pozostanie pusty, to zostanie użyty oryginalny, w przeciwnym
20     razie zostanie wyświetlony ten podany.-->
21
22     <div data-role="header">
23       <h1>Pierwsza</h1>
24     </div>
25
26     <div data-role="content">
27       <p><a href="#second">i przejście na drugą stronę</a></p>
28     </div>
29
30   </div>
```

```

31
32 <div data-role="page" id="second" data-title="Tytuł drugiej strony">
33
34   <div data-role="header">
35     <h1>Druga</h1>
36   </div>
37
38   <div data-role="content">
39     <p><a href="#first">i powrót na pierwszą stronę</a></p>
40   </div>
41 </div>
42
43 </body>
44 </html>

```

---

W 3. wierszu kodu został umieszczony domyślny element `title`. To właśnie on jest wczytywany i wyświetlany domyślnie. W wierszach 17. i 32. definiowane są natomiast tytuły konkretnych stron. Gdy tylko przejdziemy na drugą stronę, tytuł wyświetlany przez przeglądarkę zmieni się — zostanie zastąpiony zawartością atrybutu `data-title`, podaną w wierszu 32.

Po kliknięciu przycisku *Wstecz* tytuł zostanie ponownie zmieniony i zastąpiony domyślnym, podanym w wierszu 3. Aby wyświetlić tytuł odpowiadający pierwszej stronie, czyli ten podany w wierszu 17., należy utworzyć odnośnik odwołujący się do pierwszej strony. Właśnie taki odnośnik można znaleźć w 39. wierszu kodu. Wystarczy przetestować przykład, by przekonać się, jak on działa.

## Receptura: Wczytywanie zewnętrznych stron przy użyciu technologii AJAX

Aż do tej receptury wszystkie inne przykłady wyświetlały strony, które były już dostępne w kodzie dokumentu HTML. Wszystkie one działały, korzystając z etykiet odwołujących się do identyfikatorów elementów, które już istniały w drzewie DOM.

Jednak wraz z powiększaniem się witryny umieszczanie wszystkich stron w jednym dokumencie HTML stanie się niemożliwe. Dzięki jQuery Mobile tworzenie odnośników do innych stron HTML jest równie łatwe jak w zwyczajnych aplikacjach internetowych. Listing 9.4 przedstawia odnośniki, które wyglądają dokładnie tak samo.

Listing 9.4. Odnośniki do zewnętrznych stron

---

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Odnośniki - wykorzystanie technologii AJAX</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">

```



```
08 <script type="text/javascript"
09   src="http://code.jquery.com/jquery-1.7.1.min.js">
10 </script>
11 <script type="text/javascript" src=
12   "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13 </script>
14 </head>
15 <body>
16
17 <div data-role="page">
18
19   <div data-role="header">
20     <h1>Odnosińki - wykorzystanie technologii AJAX</h1>
21   </div>
22
23   <div data-role="content">
24     <p><a href="04b-link.html">Odnosińki do zewnętrznego pliku</a></p>
25     <p><a href="04b-link.html" data-prefetch>Wstępne wczytanie
26       zewnętrznego pliku</a></p>
27   </div>
28
29 </div>
30
31 </body>
32 </html>
```

---

Różnica polega na sposobie, w jaki jQuery Mobile obsługuje odnośniki. Kiedy odnośnik wskazuje stronę należącą do tej samej domeny, to domyślnie strona ta jest pobierana przy użyciu odwołania wykonywanego z wykorzystaniem technologii AJAX. Ten domyślny sposób działania można zmienić za pomocą odpowiedniego ustawienia, co pokazano w dalszej części rozdziału. Oprócz tego, jeśli odnośnik odwołuje się do strony należącej do innej domeny, to może zostać potraktowany jako odnośnik do normalnej witryny WWW.

Zaletą wczytywania treści przy użyciu technologii AJAX jest możliwość stosowania płynnych animacji przejść pomiędzy stronami. Oprócz tego, poszczególne strony są przechowywane w drzewie DOM. Pozwala to na szybkie cofanie się do poprzedniej strony, bez konieczności odwoływania się do serwera. Jednak wadą takiego rozwiązania jest zwiększone zużycie pamięci. Może się ono zatem okazać problemem na urządzeniach dysponujących ograniczonymi zasobami.

### Uwaga

Podczas tworzenia aplikacji na własnym komputerze należy używać serwera WWW. W razie tworzenia stron z użyciem jQuery Mobile bez korzystania z serwera żądania wykonywane przy użyciu technologii AJAX nie będą działały, a na ekranie bądź w konsoli będą wyświetlane błędy. Choć niektóre przeglądarki pozwalają na wczytywanie zasobów lokalnych nawet za pośrednictwem żądań wykonywanych przy użyciu technologii AJAX, to jednak zawsze warto, o ile to tylko możliwe, odtwarzać środowisko produkcyjne.

Wiersze 24. oraz 25. zawierają dwie różne wersje odnośnika do pliku zewnętrznego. Część twórców woli unikać stosowania atrybutu `data-prefetch` we wszystkich odnośnikach, gdyż powoduje on generację dodatkowego żądania HTTP, co może pociągnąć za sobą spowolnienie wyświetlania i wczytywania witryny. Można zdecydować się na stosowanie ich wyłącznie w odnośnikach do stron, które na pewno zostaną odwiedzone. Pierwszy z przedstawionych odnośników wczyta stronę, gdy zostanie kliknięty; natomiast drugi wczyta ją możliwie jak najszybciej — nawet bez klikania — by przyspieszyć poruszanie się po witrynie.

Listing 9.5 przedstawia kod wczytywanej zewnętrznej strony.

Listing 9.5. Wczytywana zewnętrzna strona

---

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Dołączona strona</title>
04   </head>
05 <body>
06
07 <div data-role="page">
08
09   <div data-role="header">
10     <h1>Dołączona strona</h1>
11   </div>
12
13   <div data-role="content">
14     <p>
15       <a href="04-linking-ajax.html">
16         Odnośnik do początkowej strony
17       </a>
18     </p>
19   </div>
20
21 </div>
22
23 </body>
24 </html>

```

---

Warto zwrócić uwagę, że w sekcji nagłówka tego pliku nie ma żadnego kodu CSS ani JavaScript. W tym przykładzie ma to służyć pokazaniu, że strona faktycznie jest wczytywana przy użyciu technologii AJAX oraz że skrypty i arkusze CSS ze strony początkowej wciąż będą dostępne i używane.

W produkcyjnej wersji aplikacji warto pomimo to umieścić w tej zewnętrznej stronie arkusze stylów i skrypty, gdyż istnieje prawdopodobieństwo, że ktoś odwoła się do niej bezpośrednio. Można spróbować to zrobić, lecz w takim przypadku strona nie będzie mieć charakterystycznego wyglądu nadawanego przez jQuery Mobile.

Szczególną uwagę należy zwrócić na zawartość paska adresu URL przeglądarki. Choć strony są wczytywane przy użyciu technologii AJAX, to adres URL zmienia się zgodnie z aktualnie prezentowaną zawartością. Daje to możliwość zapamiętywania adresów stron i ułatwia poruszanie się po witrynie.

## Receptura: Wyświetlanie komunikatów o wczytywaniu stron

Kiedy serwer, z którego korzystamy, jest wolny bądź jeśli dysponujemy wolnym połączeniem z internetem, co w przypadku urządzeń mobilnych jest znacznie bardziej prawdopodobne, to podczas wczytywania stron tworzonych przy użyciu jQuery Mobile są zazwyczaj wyświetlane komunikaty.

Informacje te mogą być używane także w innych celach, a nie tylko podczas klikania odnośników (na przykład, kiedy używany kod JavaScript będzie pobierał zawartość z serwera przy użyciu technologii AJAX bądź gdy sam będzie wykonywał złożone i czasochłonne obliczenia). Listing 9.6 pokazuje, w jaki sposób można wyświetlać komunikaty wczytywania strony.

Listing 9.6. Włączanie i wyłączanie komunikatów

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Komunikaty o wczytywaniu</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14     <script>
15       $(document).ready(function() {
16         $('#show').on('click', function() {
17           $.mobile.showPageLoadingMsg();
18         });
19         $('#hide').on('click', function() {
20           $.mobile.hidePageLoadingMsg();
21         });
22       });
23     </script>
24   </head>
25   <body>
26
27   <div data-role="page">
28
29     <div data-role="header">
30       <h1>Pokaż/ukryj komunikaty o wczytywaniu</h1>
31     </div>
32
33     <div data-role="content">
34       <a href="#" id="show" data-role="button">Pokaż komunikaty
35         o wczytywaniu</a>
36       <a href="#" id="hide" data-role="button">Ukryj komunikaty
37         o wczytywaniu</a>
```

```

38 </div>
39
40 </div>
41
42 </body>
43 </html>

```

W wierszach 34. – 37. zostały umieszczone dwa przyciski służące do włączania i wyłączenia wyświetlania komunikatów. Przyciski te zostały powiązane z napisanym przez nas kodem JavaScript. Choć zazwyczaj jQuery Mobile nie wymaga pisania takiego kodu, to jednak istnieje kilka wyjątków od tej reguły. Za włączanie i wyłączenie wyświetlania komunikatów o wczytywaniu odpowiadają wywołania umieszczone odpowiednio w wierszach 17. i 20. Korzystając z tych przycisków, można zauważyć, że właściwie nie ma żadnego powiązania pomiędzy komunikatami a operacjami, które są faktycznie wykonywane.

Choć komunikaty można włączać i wyłączać w dowolnej chwili, to jednak stosowanie ich należy ograniczać do tych przypadków, kiedy faktycznie aplikacja wykonuje jakies operacje w tle lub wczytuje zasoby.

Warto zauważyć, że wywołania o postaci `$(document).ready(function() {})` można użyć tylko raz — kod umieszczony w wewnętrznej funkcji zostanie wykonany wyłącznie po zakończeniu wczytywania kodu HTML dokumentu. Podczas wczytywania nowych stron przy użyciu technologii AJAX powyższe zdarzenie nie będzie już zgłaszane. Jeśli zależy nam na wykonywaniu jakiegoś kodu podczas wyświetlania każdej ze stron, to należy w tym celu użyć wywołania o postaci `$(document).on('pageinit', function() {})`.

## Receptura: Odwołania do zewnętrznych stron bez korzystania z technologii AJAX

Biblioteka jQuery Mobile domyślnie przekształca wszystkie odnośniki na żądania asynchroniczne wykonywane przy użyciu technologii AJAX, o ile tylko odwołują się one do stron należących do tej samej domeny. Niemniej jednak mogą się zdarzyć sytuacje, w których będziemy chcieli uniknąć takiego sposobu działania biblioteki. Może tak być na przykład w sytuacji, gdy niektóre fragmenty witryny zostaną napisane z wykorzystaniem innej platformy, która nie działa prawidłowo w przypadku wczytywania stron przy użyciu technologii AJAX. Listing 9.7 pokazuje kilka sposobów na wyłączenie wczytywania stron przy wykorzystaniu tej technologii.

Listing 9.7. Odnośniki do zewnętrznych stron, które mają być pobierane bez użycia technologii AJAX

```

00 <!DOCTYPE html>
01 <html>
02 <head>
03 <title>Odnośniki, które nie korzystają z technologii AJAX</title>
04 <meta name="viewport"
05 content="width=device-width, initial-scale=1">
06 <link rel="stylesheet" href=

```

```

07     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09     src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14     </head>
15     <body>
16
17     <div data-role="page">
18
19     <div data-role="header">
20     <h1>Odnośniki, które nie korzystają z technologii AJAX</h1>
21     </div>
22
23     <div data-role="content">
24     <p><a href="04b-link.html" data-ajax="false">Odnośnik do
25     zewnętrznego pliku</a></p>
26     <!-- LUB:
27     <p><a href="04b-link.html" rel="external">Odnośnik do
28     zewnętrznego pliku</a></p>
29     -->
30     </div>
31
32     </div>
33
34     </body>
35     </html>

```

W wierszu 24. został umieszczony atrybut `data-ajax`, określający, czy należy korzystać z technologii AJAX czy nie. Domyślną wartością tego atrybutu jest `true`. Jeśli jednak zostanie mu przypisana wartość `false`, to asynchroniczne żądania wykonywane przy użyciu technologii AJAX nie będą stosowane.

W wierszu 27. został umieszczony atrybut `rel="external"`. Jego zastosowanie daje takie same efekty — jQuery Mobile nie będzie pobierać zasobów przy użyciu technologii AJAX. Ten zapis jest zgodny z zaleceniami języka HTML. Korzystanie z niego może jednak powodować efekty uboczne. Niektórzy projektanci mogą bowiem określać inny wygląd odnośników prowadzących do zasobów zewnętrznych. Na przykład do takich odnośników może być dodawana ikona.

## Receptura: Określanie animacji przejścia

W przypadku kliknięcia odnośnika wewnętrznego, którego element docelowy jest pobierany przy użyciu technologii AJAX, jQuery Mobile będzie odtwarzać animację przejścia. Domyślnie zawartość strony będzie przesuwana w lewo.

Biblioteka jQuery Mobile udostępnia niewielką liczbę takich domyślnych animacji. Noszą one następujące nazwy: `pop`, `slidefade`, `slide`, `slideup`, `slidedown`, `fade` oraz `flip`.

Listing 9.8 pokazuje, w jaki sposób można zmienić używaną animację przejścia.

Listing 9.8. Przejście na inną stronę z wykorzystaniem animacji

---

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Animacje przejścia</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14   </head>
15   <body>
16
17   <div data-role="page">
18
19     <div data-role="header">
20       <h1>Pierwsza</h1>
21     </div>
22
23     <div data-role="content">
24       <p>Witaj, świecie <a href="#second" data-transition="pop">,
25         razem przejdźmy na drugą stronę.</a></p>
26       <!-- inne efekty przejścia: slide, slideup, slidedown, fade
27         oraz flip (flip jest obsługiwane tylko w systemie
28         Android) -->
29     </div>
30
31   </div>
32
33   <div data-role="page" id="second">
34
35     <div data-role="header">
36       <h1>Druga</h1>
37     </div>
38     <div data-role="content">
39       Witam ponownie!
40     </div>
41
42   </div>
43
44 </body>
45 </html>

```

---

W wierszu 24. został umieszczony atrybut `data-transition`. Można go także zastosować w elemencie `form`. Platforma jQuery Mobile sama obsługuje efekty przejścia.

## Receptura: Stosowanie niestandardowych efektów przejść

Jeżeli będziemy chcieli zastosować inną animację niż te, które są dostępne domyślnie, można ją stworzyć samemu. Można skorzystać z efektów przejść CSS3 (ang. *CSS3 transitions*), by tworzyć płynne, obsługiwane sprzętowo animacje, i to zarówno płaskie (2D), jak i przestrzenne (3D).

Listing 9.9 przedstawia przykład animacji, która powoduje obrócenie stron w momencie wczytywania nowej.

Listing 9.9. Przełączanie stron przy użyciu niestandardowych efektów przejść CSS3

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Niestandardowe efekty przejść</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <style>
09       .mine.in {
10         -webkit-animation-name: myslidein;
11       }
12
13       .mine.out {
14         -webkit-animation-name: myslideout;
15       }
16
17       @-webkit-keyframes myslidein {
18         from { -webkit-transform: rotateZ(0deg) scale(0); }
19         to { -webkit-transform: rotateZ(360deg) scale(1); }
20       }
21       @-webkit-keyframes myslideout {
22         from { -webkit-transform: rotateZ(360deg) scale(1); }
23         to { -webkit-transform: rotateZ(0deg) scale(0); }
24       }
25     </style>
26     <script type="text/javascript"
27       src="http://code.jquery.com/jquery-1.7.1.min.js">
28     </script>
29     <script type="text/javascript" src=
30       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
31     </script>
32   </head>
33   <body>
34
35   <div data-role="page">
36
37     <div data-role="header">
38       <h1>Pierwsza</h1>
39     </div>
40
41     <div data-role="content">
```

```

42
43     <p>Witaj, świecie <a href="#second" data-transition="mine">,
44         przejdźmy razem na drugą stronę</a></p>
45
46 </div>
47
48 </div>
49
50 <div data-role="page" id="second">
51
52     <div data-role="header">
53         <h1>Druga</h1>
54     </div>
55     <div data-role="content">
56         Witaj ponownie!
57     </div>
58
59 </div>
60
61 </body>
62 </html>

```

Także w powyższym listingu 9.9 nie znajdziemy żadnego kodu JavaScript. W wierszu 43. użyto natomiast atrybutu `data-transition`, któremu przypisana została wartość `mine`. jQuery Mobile używa tej wartości, aby dodawać atrybuty `class`, kiedy jest potrzebna animacja przejścia.

W arkuszu stylów umieszczonym w wierszach 9. – 15. podane zostały dwie definicje stylów: klasa `.mine` połączona z klasą `.in` oraz klasa `.mine` połączona z klasą `.out`. Obie te definicje odwołują się do animacji CSS3 zdefiniowanych w wierszach 17. – 24.

W wierszach 17. – 24. zapisane zostały definicje animacji CSS, które powodują, że podczas przejścia strona będzie obracana. Ze względu na użycie prefiksu `-webkit-` animacje te będą działać wyłącznie w systemie iOS oraz w przeglądarkach Safari i Chrome. Aby działały one także w innych przeglądarkach, trzeba by je powielić i użyć prefiksów `-moz-`, `-ms-`, a w przyszłości, kiedy standardy staną się odpowiednio popularne i powszechnie obsługiwane, całkowicie usunąć te prefiksy. Przykład przedstawiony na tym listingu jest przeznaczony dla telefonów iPhone oraz iPadów. Aby zdobyć więcej informacji na temat CSS3 oraz animacji, można zajrzeć na stronę <http://www.html5rocks.com/en/features/presentation>.

W razie korzystania z jQuery Mobile w przeglądarce, która nie obsługuje przestrzennych efektów przejść, efekt zostanie ograniczony do zwyczajnego wygaszenia. Można także przesłonić domyślnie używany efekt przejścia i zamiast niego zastosować jakiś inny. W tym celu należy zmodyfikować globalne ustawienia biblioteki w następujący sposób:

```
$.mobile.transitionFallbacks.slideout = "none";
```

Inną sytuacją, która może skłonić do zmiany domyślnego efektu przejścia, jest korzystanie z dużych ekranów, na których animacje mogą nie być płynne. W takim przypadku można zmienić wartość właściwości `maxTransitionWidth`, co pokazuje poniższy przykład:

```
$.mobile.maxTransitionWidth = 640;
```



Powyższa instrukcja sprawia, że jeśli szerokość ekranu będzie większa od 640 pikseli, żaden efekt przejścia nie będzie używany (zostanie mu przypisana wartość none).

## Receptura: Obsługa zdarzeń generowanych przez urządzenia przenośne

W rozdziale 4., „Odbieranie zdarzeń i odpowiadanie na nie”, zostały podane informacje o tym, w jaki sposób można przechwytywać standardowe zdarzenia generowane przez przeglądarki WWW. Jednak urządzenia przenośne są obsługiwane raczej przy użyciu gestów niż myszki. Oprócz tego, sposoby obsługi zdarzeń na różnych urządzeniach przenośnych mogą być odmienne. Biblioteka jQuery Mobile ukrywa przed nami wszystkie te rozbieżności. Listing 9.10 pokazuje, w jaki sposób można obsługiwać kilka podstawowych typów zdarzeń mobilnych, takich jak: dotknięcie, przeciągnięcie, zmiana orientacji oraz zmiana strony. Pozostałe rodzaje zdarzeń typowych dla urządzeń przenośnych zostały opisane w dalszej części rozdziału.

Listing 9.10. Obsługa zdarzeń przeciągnięcia, zmiany orientacji i innych

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Zdarzenia</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14     <script>
15     $(document).ready(function() {
16
17     $.each(['tap taphold swipe swipeleft swiperight ' +
18       'orientationchange scrollstart scrollstop pageshow ' +
19       'pagehide'].split(' '),
20       function( i, name ) {
21
22       $(document).on(name, function(event) {
23         $('#status').append('element docelowy = ' + event.target + ' ' +
24           'typ = ' + event.type + ' <br>');
25       });
26     });
27   });
28 </script>
29
30 </head>
31 <body>
```

```

32
33 <div data-role="page">
34
35   <div data-role="header">
36     <h1>Zdarzenia</h1>
37   </div>
38
39   <div data-role="content">
40     <p><a href="04b-link.html">Odnosnik do zewnetrznego pliku.</a></p>
41     <p><a href="04b-link.html" data-prefetch>Wstepne wczytanie
42       zewnetrznego pliku</a></p>
43     <p id="status">
44   </div>
45
46 </div>
47
48 </body>
49 </html>

```

Wyświetlając w przeglądarce tę przykładową stronę, można się przekonać, w jaki sposób działają zdarzenia oraz procedury ich obsługi. Warto spróbować obrócić urządzenie, dotknąć wybranego miejsca strony, przeciągnąć palcem po ekranie w różnych kierunkach i przekonać się, jakie efekty powoduje każda z tych czynności.

Tabela 9.1 zawiera listę wszystkich zdarzeń, które można obsługiwać przy wykorzystaniu jQuery Mobile.

Tabela 9.1. *Zdarzenia charakterystyczne dla jQuery Mobile*

Typ zdarzenia	Opis
tap	Użytkownik szybko dotknął ekranu urządzenia w jednym miejscu.
taphold	Użytkownik dotknął ekranu urządzenia i przytrzymał palec przez pewien czas.
swipe	Użytkownik przeciągnął palcem po ekranie urządzenia.
swipeleft	Użytkownik przesuwając palec po ekranie w lewo.
swiperight	Użytkownik przesuwając palec po ekranie w prawo.
orientationchange	Orientacja urządzenia została zmieniona z pionowej na poziomą lub odwrotnie.
scrollstart	Użytkownik zaczął przewijać zawartość ekranu.
scrollstop	Użytkownik skończył przewijać zawartość ekranu.
pageshow	Została wyświetlona nowa strona.
pagehide	Wcześniej wyświetlana strona została ukryta.
vmouseover	Emulacja zdarzenia mouseover służąca normalizacji zdarzeń związanych z dotykiem.
vmousedown	Emulacja zdarzenia mousedown służąca normalizacji zdarzeń związanych z dotykiem.

Tabela 9.1. Zdarzenia charakterystyczne dla jQuery Mobile (ciąg dalszy)

Typ zdarzenia	Opis
vmousemove	Emulacja zdarzenia mousemove służąca normalizacji zdarzeń związanych z dotykiem.
vmouseup	Emulacja zdarzenia mouseup służąca normalizacji zdarzeń związanych z dotykiem.
vclick	Emulacja zdarzenia click służąca normalizacji zdarzeń związanych z dotykiem.
vmousecancel	To zdarzenie jest generowane w przypadku, gdy zgłoszone wcześniej zdarzenie służące do normalizacji operacji związanych z myszką okazuje się być innym zdarzeniem, takim jak swipe.
pagebeforeload	To zdarzenie jest zgłaszane przed wczytaniem strony. Do procedury jego obsługi, oprócz samego obiektu zdarzenia, przekazywany jest także drugi parametr, będący obiektem o następujących właściwościach: <code>url</code> , <code>absUrl</code> , <code>dataUrl</code> , <code>deferred</code> , <code>options</code> . Są to kolejno: trzy różne rodzaje adresów URL określające wczytywany zasób, obiekt <code>deferred</code> , przy użyciu którego można zmienić używany mechanizm wczytywania zasobu, oraz obiekt <code>options</code> , zawierający parametry przekazywane do funkcji <code>loadPage()</code> .
pageload	To zdarzenie jest zgłaszane po udanym wczytaniu strony i przetworzeniu jej do postaci drzewa DOM. W jego przypadku drugi argument procedury obsługi jest podobny do obiektu przekazywanego podczas obsługi zdarzenia <code>pagebeforeload</code> ; jednak zamiast obiektu <code>deferred</code> posiada on właściwości <code>xhr</code> oraz <code>textStatus</code> . Są to odwołania do obiektu <code>XMLHttpRequest</code> oraz informacji o wyniku żądania zapisanej w formie łańcucha znaków.
pageloadfailed	To zdarzenie jest zgłaszane, kiedy nie uda się wczytać żądanej strony. W tym przypadku obiekt przekazywany jako drugi parametr zawiera wszystkie właściwości dostępne podczas obsługi zdarzeń <code>pagebeforeload</code> oraz <code>pageload</code> , a dodatkowo zawiera także właściwość <code>errorThrown</code> .
pagebeforechange	To zdarzenie jest zgłaszane możliwie jak najwcześniej przed zmianą strony. A zatem jest ono zgłaszane przed wczytaniem nowej strony. Obsługując zdarzenie <code>pagebeforechange</code> , można zapobiec zmianie strony przy użyciu wywołania <code>event.preventDefault()</code> . Drugim parametrem przekazywanym do tej funkcji jest obiekt zawierający dwie właściwości: <code>toPage</code> oraz <code>options</code> .
pagechange	To zdarzenie jest wywoływane po zmianie strony. Także w tym przypadku drugim parametrem przekazywanym do procedury obsługi zdarzenia jest obiekt zawierający dwie właściwości: <code>toPage</code> oraz <code>options</code> .
pagechangefailed	To zdarzenie jest zgłaszane, kiedy nie uda się zmienić strony. Także w tym przypadku drugim parametrem przekazywanym do procedury obsługi zdarzenia jest obiekt zawierający dwie właściwości: <code>toPage</code> oraz <code>options</code> .

Tabela 9.1. Zdarzenia charakterystyczne dla jQuery Mobile (ciąg dalszy)

Typ zdarzenia	Opis
pagebeforeshow	To zdarzenie jest zgłaszane bezpośrednio przed wyświetleniem nowej strony. W momencie zgłaszania tego zdarzenia animacja przejścia jeszcze nie została uruchomiona. Drugim parametrem procedury obsługi jest obiekt zawierający właściwość <code>prevPage</code> .
pagebeforehide	To zdarzenie jest zgłaszane bezpośrednio przed ukryciem aktualnie wyświetlanej strony. W momencie zgłaszania tego zdarzenia animacja przejścia jeszcze nie została uruchomiona. Drugim parametrem procedury obsługi jest obiekt zawierający właściwość <code>nextPage</code> .
pageshow	To zdarzenie jest przekazywane do nowej strony, bezpośrednio po zakończeniu efektu przejścia. Pod innymi względami przypomina ono zdarzenie <code>pagebeforeshow</code> .
pagehide	To zdarzenie jest przekazywane do starej strony tuż po zakończeniu efektu przejścia. Pod innymi względami przypomina ono zdarzenie <code>pagebeforehide</code> .
pagebeforecreate	To zdarzenie jest zgłaszane w momencie tworzenia nowej strony, zanim jQuery Mobile przetworzy jej kod HTML i zainicjuje widżety. Wskazówka: warto go używać, by uprościć kod HTML strony i określać wartości takich atrybutów jak <code>data-role</code> z poziomu kodu JavaScript.
pagecreate	To zdarzenie jest zgłaszane po utworzeniu strony, lecz jeszcze zanim widżety zmodyfikują jej kod HTML. Można go używać do dołączania do biblioteki własnych wtyczek.
pageinit	To zdarzenie jest zgłaszane po wczytaniu strony. Służy ono bibliotece jQuery Mobile jako zamiennik zdarzenia <code>ready</code> , w przypadkach gdy na stronach wczytywanych przy użyciu technologii AJAX jest jakiś kod, który powinien być wykonywany po ich wczytaniu. Zaleca się kojarzenie tego zdarzenia z elementami <code>div</code> reprezentującymi strony, a nie z całymi dokumentami HTML.
pageremove	To zdarzenie jest generowane bezpośrednio przed usunięciem strony.
updatelayout	To zdarzenie jest generowane w momencie wyświetlania lub ukrywania komponentu. Stanowi ono sygnał dla innych komponentów, że powinny przeliczyć swój rozmiar i położenie.

W wierszach 16. – 19. wymienionych zostało jedynie kilka spośród wszystkich dostępnych zdarzeń. Zdarzenia te są standardowo używane podczas interakcji z użytkownikami urządzeń przenośnych. Można je zastąpić innymi zdarzeniami z tabeli 9.1, by przestudiować działanie innych elementów mechanizmu obsługi zdarzeń biblioteki jQuery Mobile (na przykład: cykl wczytywania stron). Dzięki wykorzystaniu wirtualnych zdarzeń związanych z myszką można zapewnić odpowiednie działanie aplikacji zarówno na urządzeniach obsługiwanych przy użyciu dotyku, jak i komputerach wyposażonych w myszkę. W przypadku korzystania z komputera wyposażonego w myszkę jQuery Mobile automatycznie skorzysta z odpowiednich procedur, by prawidłowo obsługiwać

to urządzenie. Z kolei w razie korzystania z urządzenia obsługiwane przy użyciu dotyku zdarzenia są zamieniane na odpowiednie zdarzenia związane z dotykiem, zgłaszane i obsługiwane w takiej samej kolejności co odpowiadające im zdarzenia związane z myszką. Standaryzowane są także informacje przekazywane do procedur obsługi zdarzeń związanych z obsługą myszy i z dotykiem, dzięki czemu wszystkie zwracane współrzędne będą takie same. Podczas korzystania z wirtualnych zdarzeń myszy można zrezygnować ze stosowania zdarzenia `click`. W przeglądarkach mobilnych bazujących na silniku WebKit może ono bowiem powodować 300-milisekundowe opóźnienie, które nie tylko sprawi, że animacje nie będą płynne, lecz co gorsza może doprowadzić do pojawiania się niepożądanych zdarzeń dwukrotnego kliknięcia.

## Receptura: Generacja przycisku Wstecz

Podczas poruszania się po witrynie czasami może się przydać wyświetlanie na jej stronach przycisku pozwalającego na cofnięcie się na poprzednią stronę. Oczywiście, przeglądarki WWW udostępniają standardowy przycisk, który służy właśnie do tego celu.

Jednak można wskazać ważne powody przemawiające za umieszczeniem takiego przycisku bezpośrednio w aplikacji. Jednym z nich jest to, że kiedy aplikacja działa w trybie pełnoekranowym, przyciski przeglądarki nie są widoczne. Ważniejsze jest jednak to, że internetowe aplikacje tworzone przy użyciu biblioteki jQuery Mobile udają rodzime aplikacje komputerowe. A w nich przycisk *Wstecz* (ang. *Back*) jest umieszczony w lewym górnym wierzchołku ekranu.

Listing 9.11 pokazuje, w jaki sposób jQuery Mobile ułatwia wyświetlanie przycisku *Wstecz*, kiedy aplikacja będzie go potrzebować.

Listing 9.11. Przechodzenie na drugą stronę, gdy dostępny jest przycisk Wstecz

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Przyciski Wstecz</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14
15   </head>
16   <body>
17     <div data-role="page">
18
19       <div data-role="header">
```

```

20     <h1>Pierwsza</h1>
21   </div>
22
23   <div data-role="content">
24     <p>Przejdź <a href="#second">na drugą stronę</a>, aby
25       zobaczyć przycisk "Wstecz".</p>
26   </div>
27
28 </div>
29 <div data-role="page" id="second"
30     data-add-back-btn="true" data-back-btn-text="Wstecz">
31
32   <div data-role="header">
33     <h1>Druga</h1>
34   </div>
35
36   <div data-role="content">
37     <p>Treść drugiej strony.</p>
38   </div>
39
40 </div>
41
42 </body>
43 </html>

```

Pierwszą rzeczą, jaką można zauważyć po wyświetleniu tej strony w przeglądarce, jest **brak** przycisku *Wstecz*. Jest to całkiem słuszne, gdyż ciągle przebywamy na pierwszej stronie, więc nie ma się gdzie cofnąć. Kolejnym powodem przemawiającym za takim rozwiązaniem jest to, że na pierwszej stronie nie został zdefiniowany atrybut `data-add-back-button="true"`<sup>1</sup>. Można spróbować, co się stanie, jeśli atrybut ten zostanie dodany także do pierwszej strony aplikacji.

Po przejściu na drugą stronę przycisk *Wstecz* zostanie wyświetlony w lewym górnym wierzchołku strony. Warto zwrócić uwagę, że wiersze 34. – 36. nie zawierają jednak żadnego kodu związanego z tym przyciskiem.

Oprócz tego można także dodać atrybut `data-rel="back"` do odnośnika, co sprawi, że jego kliknięcie spowoduje wyświetlenie poprzedniej strony. W razie stosowania takiego rozwiązania w atrybucie `href` należy podać faktyczny adres URL, tak by starsze przeglądarki i urządzenia były w stanie wyświetlić odpowiednią stronę.

## Receptura: Dodawanie stopek u dołu strony

W przykładach przedstawionych do tej pory używane były jedynie nagłówki oraz elementy `div` zawierające treść strony. Jednak w bardzo podobny sposób można także tworzyć elementy `div`, które będą wyświetlane jako stopki. Korzystając ze stopek, należy

<sup>1</sup> Atrybut `data-back-btn-text` pozwala na określenie tekstu wyświetlanego na przycisku; domyślnie jest to słowo *Back* — *przyp. tłum.*

pamiętać o dwóch zagadnieniach: jakie jest ich położenie na ekranie oraz co chcemy w nich wyświetlać. Listing 9.12 przedstawia przykład strony ze stopką, w której zostało umieszczonych kilka przycisków.

Listing 9.12. Wyświetlanie u dołu strony stopki z kilkoma przyciskami

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Stopka</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14
15   </head>
16   <body>
17
18     <div data-role="page">
19
20       <div data-role="header">
21         <h1>Stopka</h1>
22       </div>
23
24       <div data-role="content">
25         <p>Zawartość strony</p>
26       </div>
27
28       <div data-role="footer">
29         <h1>Stopka</h1>
30         <a href="#">Przycisk?</a>
31         <a href="index.html" data-role="button" data-icon="delete">
32           Usuń
33         </a>
34         <a href="index.html" data-role="button" data-icon="plus">
35           Dodaj
36         </a>
37       </div>
38     </div>
39
40
41   </body>
42 </html>
```

Pierwszy przycisk został zdefiniowany w wierszu 30. Jak widać, by stworzyć przycisk, wystarczy użyć zwyczajnego elementu odnośnika. Odnośniki w stopkach są domyślnie wyświetlane w formie przycisków. Wiersze 31. – 36. zawierają dwa przyciski zdefiniowane

w standardowy sposób, przy czym do każdego z nich została dodana ikona. Listę wszystkich aktualnie dostępnych ikon można znaleźć na stronie: <http://api.jquerymobile.com/mobileicons/>.

Element `div` zawierający treść strony jest w tym przykładzie niemal pusty. Można dodać do niego nieco więcej treści i przekonać się, jak to wpłynie na położenie stopki podczas przewijania strony w górę i w dół. Stopka zostanie wyświetlona poniżej dodanej treści i będzie umieszczona u dołu strony.

Zamiast niezależnych przycisków można także stworzyć ich grupę. Domyślnie przyciski wchodzące w skład grupy przycisków są rozmieszczane w pionie. Dzięki zastosowaniu atrybutu `data-type="horizontal"` można jednak wyświetlić wszystkie przyciski w formie jednego poziomego prostokąta, którego lewa i prawa krawędź będzie zaokrąglona. Przykład takiego układu grupy przycisków został przedstawiony na listingu 9.13.

Listing 9.13. Wyświetlanie grupy przycisków w stopce

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Grupa przycisków w stopce</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14
15   </head>
16   <body>
17
18     <div data-role="page">
19
20       <div data-role="header">
21         <h1>Grupa przycisków w stopce</h1>
22       </div>
23
24       <div data-role="content">
25         <p>Zawartość strony</p>
26       </div>
27
28
29       <div data-role="footer">
30         <div data-role="controlgroup" data-type="horizontal">
31           <a href="#" data-role="button">Pierwszy</a>
32           <a href="#" data-role="button">Drugi</a>
33           <a href="#" data-role="button">Trzeci</a>
34           <a href="#" data-role="button">Czwarty</a>
35         </div>

```



```
36 </div>
37
38 </div>
39
40
41 </body>
42 </html>
```

---

W wierszach 30. – 35. umieszczony został element `div` z atrybutami `data-role="controlgroup"` oraz `data-type="horizontal"`, a wewnątrz niego grupa niezależnych przycisków.

## Receptura: Ta sama stopka na wielu stronach

Kliknięcie odnośnika powoduje uruchomienie animacji podczas zmiany wyświetlanej strony. Może się jednak zdarzyć, że nie będziemy chcieli, by stopka była objęta tą animacją. Listing 9.14 pokazuje, w jaki sposób sprawić, by stopka cały czas była widoczna podczas przechodzenia na inną stronę.

Listing 9.14. Wyświetlanie stopki podczas zmiany strony

---

```
00 <!DOCTYPE html>
01 <html>
02 <head>
03 <title>Ustalona stopka</title>
04 <meta name="viewport"
05 <content="width=device-width, initial-scale=1">
06 <link rel="stylesheet" href=
07 <"http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08 <script type="text/javascript"
09 <src="http://code.jquery.com/jquery-1.7.1.min.js">
10 </script>
11 <script type="text/javascript" src=
12 <"http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13 </script>
14
15 </head>
16 <body>
17 <div data-role="page">
18
19 <div data-role="header">
20 <h1>Pierwsza</h1>
21 </div>
22
23 <div data-role="content">
24 <p>Przejdź <a href="#second">na drugą stronę</a>.</p>
25 </div>
26
27 <div data-role="footer" data-id="myfooter" data-position="fixed">
28 <p>Pierwsza strona</p>
29 </div>
30
```

```

31 </div>
32 <div data-role="page" id="second"
33     data-add-back-btn="true" data-back-btn-text="Wstecz">
34
35 <div data-role="header">
36 <h1>Druga strona</h1>
37 </div>
38
39 <div data-role="content">
40 <p>Zawartość drugiej strony.</p>
41 <p>Więcej treści.</p>
42
43 </div>
44 <div data-role="footer" data-id="myfooter" data-position="fixed">
45 <p>Druga strona</p>
46 </div>
47
48 </div>
49
50 </body>
51 </html>

```

---

Elementy div umieszczone w wierszach 27. i 44. zawierają atrybut `data-position="fixed"`. Po kliknięciu odnośnika można się przekonać, że animowany jest nagłówek oraz treść strony, lecz nie jej stopka. Oczywiście, widać także, że zmieniła się zawartość stopki.

## Receptura: Wyświetlanie stopki w ustalonym miejscu

W wersjach jQuery Mobile wcześniejszych od 1.1 podczas przewijania strony stopka chwilowo zniknęła i pojawiała się ponownie po zakończeniu przewijania. Jednak po wprowadzeniu wersji 1.1 stopka i nagłówki pozostają u dołu i u góry strony. Jednak byłoby znacznie bardziej elegancko, gdyby nagłówek i stopka pozostawały w tym samym położeniu, a jedynie treść strony była przewijana. I właśnie taką możliwość dają ustalone paski narzędzi. Listing 9.15 przedstawia, w jaki sposób można je tworzyć i stosować.

---

Listing 9.15. Przewijanie tekstu bez modyfikacji położenia stopki

```

00 <!DOCTYPE html>
01 <html>
02 <head>
03 <title>Stałe położenie</title>
04 <meta name="viewport"
05     content="width=device-width, initial-scale=1">
06 <link rel="stylesheet" href=
07     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08 <script type="text/javascript"
09     src="http://code.jquery.com/jquery-1.7.1.min.js">
10 </script>
11 <script type="text/javascript" src=
12     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">

```

```
13 </script>
14 <script>
15 $.mobile.fixedToolbars
16   .show(true);
17   $.mobile.touchOverflowEnabled = true;
18 </script>
19
20 </head>
21 <body>
22
23 <div data-role="page">
24
25   <div data-role="header" data-position="fixed">
26     <h1>Stałe położenie</h1>
27   </div>
28
29   <div data-role="content" >
30     <p>
31       Lorem ipsum dolor sit amet, consectetur adipiscing elit. Cras
32       metus tellus, iaculis vestibulum ornare sit amet, semper ac
33       nisi. Suspendisse convallis, libero ut sodales interdum,
34       turpis ligula lacinia justo, a accumsan tellus est at lacus.
35       Morbi ultricies posuere enim, sit amet luctus massa faucibus
36       ut. Maecenas vel mi quis massa volutpat consequat ac non mi.
37       Nam et ornare sapien. Donec vitae magna sed neque lacinia
38       imperdiet. Vivamus tellus velit, molestie in interdum vel,
39       gravida vel mauris. Vivamus justo augue, ultrices ut viverra
40       ut, sollicitudin id lacus. Integer ornare massa ut risus
41       tempus lobortis. Donec ac nisi eu nunc volutpat posuere
42       dapibus ut nisi. Nam sit amet mauris a ante vehicula mattis.
43       Phasellus rutrum rutrum enim, at convallis neque convallis eu.
44       Duis dictum justo venenatis mauris feugiat quis aliquam enim
45       egestas. Integer et ante metus, ut faucibus libero.
46     </p>
47
48     <p>
49       Sed lobortis nunc nec ligula dictum dignissim pellentesque
50       lorem semper. Vivamus dui felis, pulvinar non accumsan ac,
51       facilisis a lectus. In blandit aliquet sapien sed eleifend.
52       Mauris ut arcu nisl. Morbi eget sapien vulputate lectus
53       dapibus congue. Cras id odio nulla, quis viverra massa. Mauris
54       tortor nisl, tincidunt et vestibulum nec, blandit ut purus. In
55       vel massa a erat tristique lacinia. Vestibulum malesuada
56       tristique nunc, in dictum quam faucibus a. Etiam sed enim
57       ante, et aliquam quam. Donec velit velit, cursus at sodales
58       id, accumsan at sapien.
59     </p>
60   </div>
61   <div data-role="footer" data-position="fixed">
62     <p>Druga strona</p>
63   </div>
64
65 </div>
66
67
68 </body>
69 </html>
```

---

Kod umieszczony w wierszach 15. – 16. zapewnia, że nagłówki i stopki będą cały czas wyświetlane u góry i u dołu ekranu, a nie na początku i na końcu zawartości strony. W wierszu 17. właściwość `touchOverflowEnabled` przypisywana jest wartość `true`; powoduje ona, że obszar przewijany przy użyciu pasków przewijania zostaje ograniczony do zawartości strony.

## Receptura: Wyświetlanie i ukrywanie stopki

W przypadku korzystania z normalnych stopek, które są wyświetlane i ukrywane, gdy użytkownik przewija zawartość strony, istnieje także możliwość ich wyświetlania i ukrywania z poziomu kodu JavaScript. Listing 9.16 pokazuje, jak można to zrobić.

Listing 9.16. Dodawanie przycisków ukrywających i wyświetlających stopkę

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Ustalona stopka</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14     <script>
15       $(document).ready(function() {
16         $.mobile.fixedToolbars
17           .setTouchToggleEnabled(false);
18         $.mobile.fixedToolbars
19           .show(true);
20         $.mobile.touchOverflowEnabled = true;
21         $('#show').on('click', function() {
22           $.mobile.fixedToolbars.show();
23         });
24         $('#hide').on('click', function() {
25           $.mobile.fixedToolbars.hide();
26         });
27       });
28     </script>
29 </head>
30 <body>
31
32   <div data-role="page">
33

```

```

34 <div data-role="header" data-position="fixed">
35   <h1>Ustalona stopka</h1>
36 </div>
37
38 <div data-role="content" >
39   <p>Przejdź na <a href="#second">drugą stronę</a>.</p>
40   <a href="#" id="show" data-role="button">Pokaż stopkę</a>
41   <a href="#" id="hide" data-role="button">Ukryj stopkę</a>
42
43   <p> Lorem ipsum dolor sit amet, consectetur adipiscing elit.
44   Cras metus tellus, iaculis vestibulum ornare sit amet,
45   semper ac nisi. Suspendisse convallis, libero ut sodales
46   interdum, turpis ligula lacinia justo, a accumsan tellus est
47   at lacus. Morbi ultricies posuere enim, sit amet luctus
48   massa faucibus ut. Maecenas vel mi quis massa volutpat
49   consequat ac non mi. Nam et ornare sapien. Donec vitae magna
50   sed neque lacinia imperdiet. Vivamus tellus velit, molestie
51   in interdum vel, gravida vel mauris. Vivamus justo augue,
52   ultrices ut viverra ut, sollicitudin id lacus. Integer
53   ornare massa ut risus tempus lobortis. Donec ac nisi eu nunc
54   volutpat posuere dapibus ut nisi. Nam sit amet mauris a ante
55   vehicula mattis. Phasellus rutrum rutrum enim, at convallis
56   neque convallis eu. Duis dictum justo venenatis mauris
57   feugiat quis aliquam enim egestas. Integer et ante metus, ut
58   faucibus libero. </p>
59
60   <p> Sed lobortis nunc nec ligula dictum dignissim
61   pellentesque lorem semper. Vivamus dui felis, pulvinar non
62   accumsan ac, facilisis a lectus. In blandit aliquet sapien
63   sed eleifend. Mauris ut arcu nisl. Morbi eget sapien
64   vulputate lectus dapibus congue. Cras id odio nulla, quis
65   viverra massa. Mauris tortor nisl, tincidunt et vestibulum
66   nec, blandit ut purus. In vel massa a erat tristique
67   lacinia. Vestibulum malesuada tristique nunc, in dictum quam
68   faucibus a. Etiam sed enim ante, et aliquam quam. Donec
69   velit velit, cursus at sodales id, accumsan at sapien. </p>
70
71 </div>
72 <div data-role="footer" data-position="fixed">
73   <p>Druga strona</p>
74 </div>
75
76 </div>
77
78
79 </body>
80 </html>

```

---

Wiersze od 16. – 26. zawierają kod umożliwiający wyświetlanie i chowanie nagłówków i stopki. Kiedy stopka jest „ukryta”, to wciąż jest widoczna u dołu strony — nie jest natomiast wyświetlana u dołu ekranu. To samo dotyczy nagłówków.

## Receptura: Optymalizacja nagłówków i stopek pod kątem prezentacji pełnoekranowych

jQuery Mobile udostępnia możliwość określaną jako tryb pełnoekranowy. W tym trybie nagłówek i stopka domyślnie przesłaniają treść strony, lecz kiedy użytkownik kliknie tę treść, nagłówek i stopka znikają. Jeśli zainstalujemy aplikację internetową na telefonie w formie ikony wyświetlanej wraz ze wszystkimi innymi aplikacjami bądź też jeśli zintegrujemy ją z jakąś rodzimą platformą do tworzenia aplikacji internetowych, taką jak Phonegap, to nasza aplikacja może być wyświetlana w trybie pełnoekranowym, bez widocznego paska adresu, paska statusu oraz pasków przewijania przeglądarki. Aby móc optymalnie korzystać z tego trybu pełnoekranowego, warto mieć możliwość całkowitego ukrywania nagłówków i stopek — inaczej niż w poprzednim przykładzie, w którym ukrycie stopki oznaczało jedynie wyświetlenie jej u dołu strony. Listing 9.17 pokazuje, jak umożliwić korzystanie z trybu pełnoekranowego.

Listing 9.17. Wyświetlanie nagłówków i stopek ponad zawartością w ramach trybu pełnoekranowego

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Tryb pełnoekranowy</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14   </head>
15   <body>
16
17   <div data-role="page" data-fullscreen="true">
18
19     <div data-role="header" data-position="fixed">
20       <h1>Nagłówek jest wyświetlany <em>nad</em> sekcją zawartości</h1>
21     </div>
22
23     <div data-role="content">
24       <h1>Nagłówek</h1>
25       <p>Kliknij, aby ukryć nagłówek i stopkę.</p>
26       <p>Drugie kliknięcie pozwoli je ponownie wyświetlić.</p>
27     </div>
28
29     <div data-role="footer" data-position="fixed">
30       <p>Jeśli zawartość strony będzie odpowiednio duża, to także
31         stopka zostanie wyświetlona nad zawartością.</p>
32     </div>
33

```

```
34 </div>
35
36 </body>
37 </html>
```

---

Wiersz 17. zawiera atrybut pozwalający na uruchomienie trybu pełnoekranowego. Aby optymalnie wykorzystać możliwości, jakie daje ten sposób prezentacji treści strony, warto określić jej wygląd przy użyciu niestandardowych arkuszy stylów.

## Receptura: Zmiana schematu kolorów przy użyciu tematów

W rozdziale 8., „Zmiany wyglądu”, przedstawione zostało narzędzie **ThemeRoller**, służące do zmiany wyglądu interfejsu użytkownika tworzonoego przy użyciu biblioteki jQuery UI. Podobne narzędzie udostępnia także biblioteka jQuery Mobile; jest ono dostępne na stronie <http://jquerymobile.com/themeroller/>.

Po zdefiniowaniu swojego własnego schematu kolorów można go stosować, określając w kodzie HTML, który temat (ang. *theme*) ma być używany. Nawet jeśli nie zdefiniujemy własnego zestawu kolorów, to wciąż możemy korzystać z kilku domyślnych, przedstawionych na listingu 9.18.

Listing 9.18. Stosowanie kilku tematów na jednej stronie

---

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Tematy - 1</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10   </script>
11   <script type="text/javascript" src=
12     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13   </script>
14
15 </head>
16 <body>
17
18 <div data-role="page">
19
20   <div data-role="header" data-theme="b">
21     <h1>Tematy - 1</h1>
22
23     <a href="#first" data-icon="arrow-l">Pierwszy</a>
24     <a href="#second" data-icon="arrow-r" data-theme="a">Drugi</a>
25
```

```

26 </div>
27 <div data-role="content">
28   <p>Treść</p>
29   <a href="#third" data-role="button" data-icon="arrow-u"
30     data-theme="c">Trzeci</a>
31   <a href="#third" data-role="button" data-icon="delete"
32     data-theme="d">Czwarty</a>
33   <a href="#third" data-role="button" data-icon="arrow-d"
34     data-theme="e">Piąty</a>
35 </div>
36
37 <div data-role="footer" data-position="fixed" data-theme="c">
38   <div data-role="controlgroup" data-type="horizontal">
39     <a href="#" data-role="button" data-theme="a">Pierwszy</a>
40     <a href="#" data-role="button" data-theme="b">Drugi</a>
41     <a href="#" data-role="button" data-theme="c">Trzeci</a>
42     <a href="#" data-role="button" data-theme="d">Czwarty</a>
43     <a href="#" data-role="button" data-theme="e">Piąty</a>
44   </div>
45 </div>
46 </div>
47
48
49 </body>
50 </html>

```

W wierszu 20. zmieniany jest temat używany w nagłówku strony. Jednak wewnątrz tego nagłówka przycisk umieszczony w wierszu 24. został wyświetlony przy użyciu innego tematu. Podobnie jest w przypadku sekcji treści strony, która używa tematu domyślnego, oprócz przycisków umieszczonych w wierszach 29. – 34., które są wyświetlane w inny sposób. Nawet zgrupowane przyciski mogą być prezentowane w odmienny sposób, co pokazuje kod zapisany w wierszach 39. – 43.

Zamiast stosować różne tematy w poszczególnych elementach, można także wybrać jeden, który będzie używany na całej stronie. Sposób zmiany używanego tematu został przedstawiony na listingu 9.19.

Listing 9.19. Zmiana tematu używanego na całej stronie

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Tematy - 2</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14

```



```

15 </head>
16 <body>
17
18 <div data-role="page" data-theme="a">
19
20   <div data-role="header" >
21     <h1>Tematy - 2</h1>
22   </div>
23
24   <div data-role="content">
25     <h1>Nagłówek</h1>
26     <p>Można także <a href="#">wybierać temat</a> określający
27       wygląd zawartości strony.</p>
28     <ul>
29       <li>0to test.</li>
30     </ul>
31   </div>
32
33 </div>
34
35
36 </body>
37 </html>

```

W tym przykładzie temat stosowany w obrębie całej strony został określony w wierszu 18. Wyświetlając stronę, można się przekonać, że nawet tło jej treści jest ciemne, a czcionka jasna.

## Receptura: Tworzenie wielu kolumn

Do tej pory przedstawione zostały sposoby tworzenia całych stron, nagłówków i stopek oraz stosowanie tematów. Jednak może się zdarzyć, że na stronie będziemy potrzebowali utworzyć większą liczbę kolumn. Zwłaszcza w przypadku stosowania tabletów może się okazać, że w celu bardziej optymalnego wykorzystania przestrzeni warto podzielić stronę na kolumny. jQuery Mobile udostępnia domyślne klasy reprezentujące odrębne kolumny. Listing 9.20 korzysta z nich, by wyświetlić wiersze z kilkoma przyciskami.

Listing 9.20. Przedstawienie kolumn na przykładzie rozmieszczania przycisków

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Układ</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">

```

```
10     </script>
11     <script type="text/javascript" src=
12         "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14
15 </head>
16 <body>
17
18 <div data-role="page">
19
20   <div data-role="header">
21     <h1>Układy</h1>
22   </div>
23
24   <div data-role="content">
25
26     <div class="ui-grid-a">
27       <div class="ui-block-a">
28         <a href="#" data-role="button">Z lewej</a>
29       </div>
30       <div class="ui-block-c">
31         <a href="#" data-role="button">Z prawej</a>
32       </div>
33     </div>
34
35     <div class="ui-grid-b">
36       <div class="ui-block-a">
37         <a href="#" data-role="button">Z lewej</a>
38       </div>
39       <div class="ui-block-b">
40         <a href="#" data-role="button">Po środku</a>
41       </div>
42       <div class="ui-block-c">
43         <a href="#" data-role="button">Z prawej</a>
44       </div>
45     </div>
46
47     <div class="ui-grid-c">
48       <div class="ui-block-a">
49         <a href="#" data-role="button">Z lewej</a>
50       </div>
51       <div class="ui-block-b">
52         <a href="#" data-role="button">Nieco z lewej</a>
53       </div>
54       <div class="ui-block-c">
55         <a href="#" data-role="button">Nieco z prawej</a>
56       </div>
57       <div class="ui-block-d">
58         <a href="#" data-role="button">Z prawej</a>
59       </div>
60     </div>
61   </div>
62 </body>
63 </html>
```

---

W pierwszej kolejności warto zwrócić uwagę na wiersze 26., 35. oraz 47. Są w nich umieszczone elementy `div` z atrybutami `class` o wartościach: `ui-grid-a`, `ui-grid-b` i `ui-grid-c`. Te klasy przekształcają elementy `div` w pojemniki zawierające wiele kolumn. Oznaczają one odpowiednio: dwie, trzy oraz cztery kolumny.

Wewnątrz tych elementów `div` umieszczone zostały zagnieżdżone elementy `div`, w których użyto klas: `ui-block-a`, `ui-block-b`, `ui-block-c` oraz `ui-block-d`. Reprezentują one poszczególne kolumny. Nazwy kolumn mogą być stosowane niezależnie od pojemnika, w którym dana kolumna została umieszczona. Oczywiście, lepiej unikać stosowania klasy `ui-block-c` wewnątrz pojemnika `ui-grid-a`, gdyż reprezentuje element zawierający jedynie dwie kolumny.

## Receptura: Zmiana stron przy wykorzystaniu skryptów

Na początku tego rozdziału przedstawione zostały receptury, które tworzyły odwołania pomiędzy stronami przy użyciu zwyczajnych odnośników HTML (`<a href="...">`). Jednak jQuery Mobile jest platformą stworzoną przy użyciu języka JavaScript. I chociaż w aplikacjach jQuery Mobile trzeba stosować bardzo niewiele kodu JavaScript, to jednak i tak może się zdarzyć, że będziemy potrzebowali możliwości zmiany strony z poziomu kodu JavaScript. Listing 9.21 pokazuje, jak można to zrobić.

Listing 9.21. Zmiana stron z poziomu kodu JavaScript

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Zmianianie stron</title>
04     <meta name="viewport"
05       content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09       src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12       "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14     <script>
15       $(document).ready(function() {
16
17         $('#change').on('click', function(event) {
18           $.mobile.changePage('43b-change-page.html',
19             {transition: 'fade'});
20         });
21
22       });
23     </script>
24 </head>
25 <body>
```

```

26
27 <div data-role="page">
28
29   <div data-role="header">
30     <h1>Zmianianie stron</h1>
31   </div>
32
33   <div data-role="content">
34     <a href="#" id="change" data-role="button">Zmień stronę</a>
35   </div>
36
37 </body>
38 </html>

```

W wierszach 18. i 19. zostało umieszczone wywołanie funkcji `changePage`. Oprócz podania adresu URL strony, którą chcemy wyświetlić, wywołanie to określa także, że strona ma być zmieniona przy użyciu efektu przejścia `fade`, a nie domyślnego. Oprócz efektu przejścia w ten sam sposób można określać także inne parametry działania jQuery Mobile.

Opcje, które można przekazywać do funkcji `changePage()`, zostały przedstawione w tabeli 9.2.

Tabela 9.2. *Dodatkowe opcje funkcji `changePage()`*

Nazwa opcji	Opis
<code>allowSamePageTransition</code>	Przypisanie wartości <code>true</code> tej opcji sprawi, że efekt przejścia zostanie odtworzony nawet w przypadku wyświetlenia tej samej strony.
<code>changeHash</code>	W razie przypisania tej opcji wartości <code>false</code> nie będzie możliwa zmiana wartości podanej w pasku adresu za znakiem <code>#</code> .
<code>Data</code>	Ta właściwość pozwala określać dodatkowe parametry, które zostaną dodane do adresu URL podczas pobierania strony przy użyciu technologii AJAX.
<code>dataUrl</code>	Określa adres URL, który po zmianie strony zostanie przypisany właściwości <code>location</code> przeglądarki.
<code>pageContainer</code>	Ta opcja pozwala zmienić miejsce, w którym nowa strona zostanie umieszczona w drzewie DOM.
<code>reloadPage</code>	Przypisanie tej opcji wartości <code>true</code> powoduje, że po zmianie strony zostanie ona ponownie wczytana.
<code>reverse</code>	Przypisanie tej opcji wartości <code>true</code> powoduje, że efekt przejścia zostanie odtworzony w przeciwnym kierunku.
<code>showLoadMsg</code>	Przypisanie tej opcji wartości <code>false</code> sprawi, że podczas zmiany strony nie będą wyświetlane komunikaty wczytywania.
<code>role</code>	Ta opcja pozwala zmienić używaną rolę. Odpowiada ona użyciu atrybutu <code>data-role</code> w elemencie odnośnika. Można jej użyć na przykład do wyświetlenia okna dialogowego.
<code>transition</code>	Ta opcja pozwala zmienić używany efekt przejścia (została ona użyta w ostatnim przykładzie).
<code>type</code>	Zmienia używaną metodę HTTP na <code>get</code> lub <code>post</code> .

Listing 9.22 przedstawia zawartość, która ma zostać wczytana i wyświetlona.

Listing 9.22. Zawartość wczytywana przy użyciu wywołania metody JavaScript

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Zmiana strony</title>
04   </head>
05   <body>
06
07     <div data-role="page">
08
09       <div data-role="header">
10         <h1>Druga strona</h1>
11       </div>
12
13       <div data-role="content">
14         <p><a href="43-change-page.html">Przejdź na poprzednią
15           stronę</a>.</p>
16       </div>
17
18     </div>
19
20   </body>
21 </html>
```

Podobnie jak w poprzednich przykładach, także w tym celowo pominięto wszystkie skrypty i style, by pokazać, że strona zostaje wczytana przy użyciu technologii AJAX. W rzeczywistych aplikacjach te style i skrypty zostałyby zapewne określone, gdyż stronę można by wyświetlać bezpośrednio poprzez podanie jej adresu URL. W przypadku wczytywania stron przy użyciu jQuery Mobile wszelkie skrypty umieszczone w sekcji nagłówka strony zostają pominięte.

## Receptura: Wczytywanie stron przy użyciu skryptów

W przypadku zmiany wyświetlanej strony najpierw zostanie ona dodana do drzewa DOM, a dopiero potem przeglądarka mobilna ją wyświetli. Dodanie do elementu odnośnika atrybutu `data-prefatch` spowoduje, że jQuery Mobile wczyta stronę z wyprzedzeniem, jeszcze zanim użytkownik kliknie odnośnik.

Podobny sposób działania jQuery Mobile można uzyskać nawet bez tworzenia odnośników w kodzie HTML. Listing 9.23 pokazuje, w jaki sposób wczytywać dane z wyprzedzeniem, z poziomu kodu JavaScript.

Listing 9.23. Użycie kodu JavaScript do wczytania strony

```
00 <!DOCTYPE html>
01 <html>
```

```

02 <head>
03   <title>Wczytywanie strony</title>
04   <meta name="viewport"
05     content="width=device-width, initial-scale=1">
06   <link rel="stylesheet" href=
07     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08   <script type="text/javascript"
09     src="http://code.jquery.com/jquery-1.7.1.min.js">
10   </script>
11   <script type="text/javascript" src=
12     "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13   </script>
14   <script>
15     $(document).ready(function() {
16
17       $('#change').on('click', function(event) {
18         $.mobile.loadPage('43b-change-page.html',
19           {transition: 'fade'})
20         .done(function() {
21           alert('Już wczytano!');
22         });
23       });
24
25     });
26   </script>
27 </head>
28 <body>
29
30 <div data-role="page">
31
32   <div data-role="header">
33     <h1>Wczytywanie strony</h1>
34   </div>
35
36   <div data-role="content">
37     <a href="#" id="change" data-role="button">Zmień stronę</a>
38   </div>
39
40 </body>
41 </html>

```

Aby obejrzeć wyniki operacji wczytywania, konieczne będzie odpowiednie narzędzie, takie jak Firebug w przeglądarce Firefox lub podobne narzędzie programistyczne w innej przeglądarce. W drzewie DOM dokumentu nowa strona pojawi się po kliknięciu przycisku, z którym został skojarzony kod JavaScript.

Kod umieszczony w wierszach 20. – 22. pokazuje, w jaki sposób metoda `loadPage` pozwala na korzystanie z obiektu obietnicy. Przedstawione wywołanie `loadPage()` najpierw określa adres strony, którą należy wczytać, a następnie określa także efekt przejścia, którego należy użyć podczas wyświetlania nowej strony. Pełne informacje o opcjach i argumentach tej metody można znaleźć w dokumentacji jQuery Mobile, na stronie <http://api.jquerymobile.com/jquery.mobile.loadPage/>. Podobnie jak w przypadku funkcji opisanych w rozdziale 5., „Komunikacja z serwerem”, także i tutaj po wczytaniu strony jest wykonywana funkcja `done()`.

## Receptura: Dołączanie danych do węzłów DOM przy użyciu jQuery Mobile

W rozdziale 1. została przedstawiona funkcja `data()`. Pozwala ona na wydajne zapisywanie danych skojarzonych z konkretnymi węzłami drzewa DOM. W tym przypadku słowo „wydajne” oznacza, że w trakcie wykonywania tych operacji nie są wprowadzane żadne zmiany w drzewie DOM.

Biblioteka jQuery Mobile udostępnia inną wersję tej funkcji: `jqmData()`. Listing 9.24 pokazuje, w jaki sposób można jej używać tak samo jak funkcji `data()`. Jedyna różnica pomiędzy nimi polega na tym, że w nazwie funkcji `jqmData()` została umieszczona przestrzeń nazw. W bibliotece jQuery Mobile przestrzenie nazw mają duże znaczenie, gdyż zabezpieczają jej kod przed wtyczkami oraz innymi bibliotekami, z których możemy korzystać oprócz jQuery Mobile. Zespół zajmujący się rozwojem biblioteki zaleca, by w przypadku korzystania z nich stosować właśnie funkcję `jqmData()`, a nie `data()`.

Listing 9.24. Dodawanie, pobieranie oraz usuwanie danych skojarzonych z węzłami DOM przy użyciu jQuery Mobile

```
00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Funkcje obsługi danych biblioteki jQuery Mobile</title>
04     <meta name="viewport"
05           content="width=device-width, initial-scale=1">
06     <link rel="stylesheet" href=
07           "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08     <script type="text/javascript"
09           src="http://code.jquery.com/jquery-1.7.1.min.js">
10     </script>
11     <script type="text/javascript" src=
12           "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13     </script>
14     <script>
15     $(document).ready(function() {
16
17         $('#setdata').on('click', function(event) {
18             $('#mydata').jqmData('mykey', 'moja wartość');
19         });
20         $('#getdata').on('click', function(event) {
21             alert($('#mydata').jqmData('mykey'));
22         });
23         $('#removedata').on('click', function(event) {
24             $('#mydata').jqmRemoveData('mykey');
25         });
26         $('#select').on('click', function(event) {
27             alert('Liczba przycisków = ' +
28                 $('#a:jqmData(role="button)').length);
29         });
30
31     });
32 </script>
33 </head>
```

```

34 <body>
35
36 <div data-role="page">
37
38   <div data-role="header">
39     <h1>Funkcje obsługi danych biblioteki jQuery Mobile</h1>
40   </div>
41
42   <div data-role="content">
43     <p id="mydata">Ten akapit służy jako pojemnik na dane.</p>
44     <a href="#" id="setdata" data-role="button">Zapis danych</a>
45     <a href="#" id="getdata" data-role="button">Pobranie danych</a>
46     <a href="#" id="removedata" data-role="button">Usunięcie danych</a>
47     <a href="#" id="select" data-role="button">Wybór</a>
48   </div>
49
50 </body>
51 </html>

```

Wiersze 18., 21. oraz 24. pokazują, jak zapisywać, pobierać oraz usuwać dane, korzystając z funkcji `jQMData()`. Porównując je z analogicznymi przykładami zamieszczonymi w rozdziale 1., łatwo zauważyć, że jest pomiędzy nimi więcej podobieństw niż różnic.

Podobnie jak `data()` także funkcja `jQMData()` odczytuje atrybuty `data-` z kodu HTML. Jest to bardzo użyteczne, gdyż pozwala na wybieranie elementów z drzewa DOM. Kod umieszczony w wierszu 28. pokazuje, w jaki sposób można używać funkcji `jQMData()` w selektorze CSS. Jak widać, wymaga to użycia nieco dłuższego kodu niż w przypadku użycia zwyczajnego selektora: `$('a[data-role="button"]')`. Jednak takie wywołanie zapewnia odpowiednie użycie przestrzeni nazw.

Przestrzeń nazw można zmienić przy użyciu opcji konfiguracyjnej `$.mobile.ns`. Jeśli użyjemy przestrzeni nazw `myns`, to musimy także odpowiednio zmienić role przycisków, używając w tym celu atrybutów `data-myns-role`.

## Receptura: Korzystanie z funkcji pomocniczych jQuery Mobile

Do realizacji żądań asynchronicznych jQuery Mobile używa kilku funkcji pomocniczych, jednak fakt korzystania z nich zazwyczaj nie jest widoczny dla użytkownika biblioteki. Dla wygody programistów z funkcji tych można także korzystać we własnym kodzie JavaScript. Listing 9.25 pokazuje, w jaki sposób można korzystać z funkcji pomocniczych służących do operowania na adresach URL.

Listing 9.25. Odczyt adresów URL przy użyciu funkcji `parseUrl()`

```

00 <!DOCTYPE html>
01 <html>
02   <head>
03     <title>Funkcje pomocnicze do przetwarzania adresów URL</title>

```



```
04 <meta name="viewport"
05   content="width=device-width, initial-scale=1">
06 <link rel="stylesheet" href=
07   "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.css">
08 <script type="text/javascript"
09   src="http://code.jquery.com/jquery-1.7.1.min.js">
10 </script>
11 <script type="text/javascript" src=
12   "http://code.jquery.com/mobile/1.1.0/jquery.mobile-1.1.0.min.js">
13 </script>
14 <script>
15 $(document).ready(function() {
16
17   $('#parse').on('click', function() {
18
19     var url = 'http://user:password@www.pearsonhighered.com' +
20             ':80/educator/series/Developers-Library' +
21             '/10483.page?key=value#first-id';
22
23     var parsedUrl =
24       JSON.stringify(
25         $.mobile.path.parseUrl(url)
26       )
27       .replace(/,/g, '<br>');
28     $('#output').html(parsedUrl);
29   });
30   $('#absolutePath').on('click', function() {
31
32     $('#output').html(
33       $.mobile.path.makePathAbsolute(
34         'nowyplik.html',
35         '/glowny/sciezka/staryplik.html'
36       )
37     );
38   });
39   $('#absoluteurl').on('click', function() {
40
41     $('#output').html(
42       $.mobile.path.makeUrlAbsolute(
43         'nowyplik.html',
44         'http://www.domena.com.pl/glowny/sciezka/staryplik.html'
45       )
46     );
47   });
48   $('#isabsolute').on('click', function() {
49
50     $('#output').html('isAbsolutePath=' +
51       $.mobile.path.isAbsolutePath(
52         'http://www.domena.com.pl/glowny/sciezka/staryplik.html'
53       )
54     );
55   });
56   $('#isrelative').on('click', function() {
57
58     $('#output').html('isRelativeUrl=' +
59       $.mobile.path.isRelativeUrl(
60         'http://www.domena.com.pl/glowny/sciezka/staryplik.html'
```

```

61     )
62     );
63   });
64   $('#samedomain').on('click', function() {
65
66     $('#output').html('isSameDomain=' +
67       $.mobile.path.isSameDomain(
68         'http://www.domena.com.pl/glowny/sciezka/staryplik.html',
69         'http://www.domena.com.pl/glowny/sciezka/nowyplik.html'
70       )
71     );
72   });
73 });
74 </script>
75 </head>
76 <body>
77
78 <div data-role="page">
79
80 <div data-role="header">
81   <h1>Funkcje pomocnicze do przetwarzania adresów URL</h1>
82 </div>
83
84 <div data-role="content">
85   <a href="#" id="parse" data-role="button">Przetworzenie
86     adresu URL</a>
87   <a href="#" id="absolutePath" data-role="button">Ścieżka
88     bezwzględna</a>
89   <a href="#" id="absoluteurl" data-role="button">Bezwzględny
90     adres URL</a>
91   <a href="#" id="isabsolute" data-role="button">Czy adres jest
92     bezwzględny?</a>
93   <a href="#" id="isrelative" data-role="button">Czy adres jest
94     względny?</a>
95   <a href="#" id="samedomain" data-role="button">Czy to ta
96     sama domena?</a>
97
98   <p id="output">Miejsce na wyniki</p>
99 </div>
100
101 </div>
102
103 </body>
104 </html>

```

W powyższym przykładzie zostało wykorzystanych kilka różnych funkcji pomocniczych. Poniżej każda z nich została opisana nieco bardziej szczegółowo.

W wierszu 25. zostało umieszczone wywołanie funkcji `parseUrl()`. Jej przeznaczeniem jest ułatwienie odczytu różnych elementów adresu URL. Funkcja ta zwraca obiekt. W tym przykładzie obiekt ten jest przekształcany do postaci łańcucha znaków zapisanego w formacie JSON, przy czym, w celu poprawienia przejrzystości wyników, po każdym przecinku jest dodawany znak nowego wiersza. Funkcja `parseUrl()` zwraca informacje przedstawione w tabeli 9.3.

Tabela 9.3. Właściwości obiektu zwracanego przez funkcję `parseUrl()`

Nazwa właściwości	Opis
<code>hash</code>	Fragment adresu URL poprzedzony znakiem <code>#</code> .
<code>host</code>	Nazwa hosta oraz numer portu podane w adresie URL.
<code>hostname</code>	Nazwa hosta podana w adresie URL, wraz z prefiksem i końcówką.
<code>href</code>	Oryginalna wartość przekazana do funkcji jako adres URL.
<code>pathname</code>	Ścieżka do pliku lub katalogu przekazana jako adres URL.
<code>port</code>	Numer portu podany w adresie URL. Jeśli numer ten nie został podany, to właściwość będzie pusta.
<code>protocol</code>	Protokół określony w adresie URL.
<code>search</code>	Łańcuch zapytania podany w adresie URL, zawiera początkowy znak <code>?</code> .
<code>authority</code>	Nazwa użytkownika, hasło oraz nazwa hosta podane w adresie URL.
<code>directory</code>	Podobna do właściwości <code>pathname</code> , lecz zawiera wyłącznie katalogi podane w adresie URL.
<code>domain</code>	Te same informacje co właściwość <code>authority</code> , uzupełnione o nazwę protokołu oraz numer portu.
<code>filename</code>	Nazwa żadanego pliku podana w adresie URL.
<code>hrefNoHash</code>	Oryginalny adres URL bez fragmentu podanego po znaku <code>#</code> .
<code>hrefNoSearch</code>	Oryginalny adres URL bez fragmentu podanego po znaku <code>?</code> .
<code>password</code>	Hasło przekazane w adresie URL.
<code>username</code>	Nazwa użytkownika przekazana w adresie URL.

W wierszach 33. – 36. zostało umieszczone wywołanie funkcji `makeAbsolutePath()`. Jeśli dysponujemy względną ścieżką dostępu — samą nazwą pliku, katalogu bądź ich kombinacją, która ewentualnie może się także zaczynać od symbolu `../` — to ta funkcja pozwala połączyć je ze ścieżką bezwzględną w celu wygenerowania ścieżki bezwzględnej.

Podobnie działa kod umieszczony w wierszach 42. – 45., choć umieszczone w nich wywołanie funkcji `makeUrlAbsolute()` generuje bezwzględny adres URL. Wywołanie funkcji `isAbsolutePath()` umieszczone w wierszach 51. – 53. pozwala sprawdzić, czy podany adres URL jest bezwzględny. Wiersze 59. – 61. zawierają wywołanie funkcji `isRelativeUrl()`, która sprawdza, czy podany adres jest względny.

W wierszach 67. – 70. zostało umieszczone wywołanie funkcji `isSameDomain()`, która sprawdza, czy dwa podane adresy URL odwołują się do tej samej domeny. Funkcja ta jest bardzo przydatna, kiedy trzeba sprawdzić, czy można skorzystać z technologii AJAX, by odwołać się do danego zasobu. Funkcja ta porównuje protokoły oraz nazwy domen dwóch przekazanych adresów URL i na ich podstawie określa, czy odwołują się one do tej samej domeny. Funkcja sprawdza także nazwy domen podrzędnych. Oznacza to, że jeśli użyjemy jej do porównania adresów `http://foo.bar.com/foo` oraz `http://www.bar.com/foo`, to zwróci ona wartość `false`.

## Podsumowanie

Ten rozdział rozpoczął się od prostego przedstawienia biblioteki jQuery Mobile. Ta część jQuery różni się od bibliotek jQuery Core oraz jQuery UI. Wymaga ona zmiany używanego kodu HTML i nadania mu takiej postaci, którą można łatwo przetwarzać przy użyciu platformy napisanej w języku JavaScript.

Po wprowadzeniu przedstawione zostały podstawowe mechanizmy poruszania się po stronach, w tym także animacje przejść oraz niewidoczne dla programistów wykorzystanie żądań asynchronicznych wykonywanych przy użyciu technologii AJAX. Następnie zostały opisane podstawowe elementy stron, takie jak nagłówki i stopki. Przedstawiono także kilka opcji pozwalających na modyfikację sposobu działania tych nagłówków i stopek. Można je wyświetlać w określonym, niezmiennym miejscu, a nawet nie pozwalać na ich ukrywanie podczas przechodzenia na inne strony. W dowolnym momencie można także je ukryć.

W końcowej części rozdziału przedstawione zostały sposoby pisania własnych skryptów JavaScript korzystających z możliwości jQuery Mobile i modyfikujących jej działanie.

# Skorowidz

## A

- adres URL, 226
- AJAX, 18, 115
- akapit, 25, 178
- akordeon, 153
  - metody, 158
  - obsługa zdarzeń, 157
  - określanie opcji, 155
  - określanie wyglądu, 154
- algorytm Bresenhama, 89
- animacja, 149
  - dodawania klas CSS, 208
  - kolorów, 204
  - prostokąta, 202
  - przejścia, 229, 232
- animacje
  - CSS3, 232
  - domyślne, 229
  - umieszczone w kolejce, 215
- aplikacje sieciowe, 36
- arkusz stylów CSS, 123
- atrybut, 65
  - class, 25, 232, 251
  - data-ajax, 229
  - data-back-btn-text, 238
  - data-dismiss, 343
  - data-filter, 297
  - data-iconpos, 268
  - data-inset, 289
  - data-items, 352
  - data-myattribute, 35
  - data-myns-role, 256
  - data-prefetch, 253

- data-prefetch, 226
- data-provide, 352
- data-role, 271, 282
- data-slide, 351
- data-source, 352
- data-split-icon, 291
- data-spy, 336
- data-title, 224
- data-toggle, 338
- data-transition, 230
- data-type, 282
- href, 238, 348
- id, 25, 222, 348
- lang, 65
- name, 75
- placeholder, 272
- rel, 65, 341
- style, 125
- type, 276

- atrybuty niestandardowe, 35
- automatyczne uzupełnianie, 159, 351
  - obsługa zdarzeń, 161
  - określanie wyglądu, 161
  - określanie opcji, 162
  - wywoływanie metod, 164

## B

- biblioteka
  - jQuery, 18
  - jQuery Core, 202, 208
  - jQuery Mobile, 19, 219
  - jQuery UI, 18
- blokowanie funkcji zwrotnej, 315

błąd  
404, 103  
obsługi żądania, 107  
błędy  
HTTP, 109  
serwera, 107

## C

CDN, Content Delivery Network, 25, 123  
CSS, 18, 124  
czas oczekiwania na odpowiedź, 111  
czynności wykonywane na wtyczce, 310

## D

dane  
JSON, 103  
JSONP, 117  
skojarzone z węzłami, 255  
tekstowe, 271  
XML, 115  
definiowanie wtyczki, 302  
dodawanie  
efektów graficznych, 206  
elementów interaktywnych, 123  
funkcji do prototypu, 303  
funkcji zwrotnych, 323  
klas, 61  
klas CSS, 208  
kodu HTML, 68  
menu do stopki, 264  
paska nawigacyjnego, 265  
podpowiedzi, 338  
stopek, 238  
treści, 70  
wyniku funkcji, 68  
zawartości, 67  
dokumentacja  
API jQuery, 20  
jQuery, 45  
jQuery Mobile, 20  
jQuery UI, 20  
Zurb Foundation, 20

dołączanie  
elementów, 72  
funkcji, 302  
wtyczki, 303  
DOM, 35  
dostęp do elementów HTML, 31  
drzewo  
DOM, 35, 73, 225, 255  
HTML, 38  
działanie selektora hover, 54  
dzielenie tekstu, 57  
dziennik zdarzeń, 116

## E

efekty  
animacji, 149  
graficzne, 206  
niestandardowe przejść, 231  
przejść CSS3, 231  
element  
a, 348  
body, 336  
button, 165  
canvas, 87  
div, 54, 105, 153, 240, 350  
fieldset, 282, 286  
img, 294  
input, 276  
li, 49, 94  
script, 118  
select, 142, 265, 280, 284  
span, 47, 48  
title, 223  
ul, 291, 338  
elementy  
DOM, 68  
formularzy, 271, 297  
HTML  
atrybuty, 65  
dodawanie kodu, 68  
kopiowanie, 73  
przeciąganie, 124  
przenoszenie, 73  
upuszczanie, 130

- usuwanie, 78
- właściwości, 65
- właściwość innerHTML, 66
- zagnieżdżanie, 80
- zmienianie kolejności, 76, 135
- zmienianie wielkości, 148
- interaktywne, 123
- listy, 292–297
- noscript, 63
- o zmiennej wielkości, 148–151
  - opcje, 149
  - przechwytywanie zdarzeń, 150
  - style, 149
  - wywoływanie metod, 151
- podrzędne HTML, 71
- sortowalne, 135–141
  - opcje, 137
  - przechwytywanie zdarzeń, 136
  - style, 136
  - wywoływanie metod, 140
- tablicy, 208
- zwijane, 268, 270
- emulacja selektora hover, 54

## F

### filtrowanie

- elementów, 51
- listy, 296

format JSON, 39, 103, 115

### formularz

- serializacja zawartości, 43
- weryfikacja danych, 113

### funkcja

- \$, 26
- \$.each, 208
- add(), 47
- addClass(), 61
- after(), 76
- ajax(), 106, 109
- alert(), 41, 330
- always(), 106, 321–324
- animate, 205
- append(), 67, 68
- appendTo(), 67, 72

- apply(), 311, 317, 325
- attr(), 65
- autocomplete(), 160
- before(), 76
- bind(), 87
- carousel(), 351
- changePage, 252
- changePage(), 252
- children(), 50
- click(), 343
- clone(), 73
- contains(), 55
- css(), 26, 57, 305
- data(), 36, 219, 255
- delegate(), 95, 96
- dequeue, 215
- detach(), 72, 78
- die(), 93
- disable(), 315
- document.getElementById(), 53
- done(), 106, 110, 254, 321
- each(), 28
- effect(), 208
- empty(), 78
- end(), 48
- extend(), 41
- extension(), 55
- fail(), 106–109, 112, 321
- filter(), 48, 53
- find(), 49
- fireWith(), 316, 325
- get(), 32, 103
- getArray(), 32
- grep(), 33
- has(), 51
- helper(), 311
- hover(), 55
- html(), 26, 57, 66, 99
- inArray(), 41
- index(), 32
- insertAfter(), 76
- insertBefore(), 76
- is(), 51
- isAbsoluteUrl(), 259
- isEmptyObject(), 41

isFunction(), 41  
 isPlainObject(), 41  
 isRelativeUrl(), 259  
 isSameDomain(), 259  
 jqmData(), 219, 255  
 length(), 34  
 live(), 93  
 load(), 104  
 loadPage(), 254  
 lock(), 315  
 makeAbsolutePath(), 259  
 makeArray(), 32  
 makeUrlAbsolute(), 259  
 map(), 30, 57  
 notify(), 323  
 notifyWith(), 326  
 off(), 96  
 offset(), 89  
 on(), 96, 98  
 parseUrl(), 256  
 position(), 89  
 prepend(), 70  
 prependTo(), 70  
 progress(), 323  
 promise(), 214, 319, 321  
 prop(), 65  
 proxy(), 98  
 querySelectorAll(), 52  
 queue(), 215  
 reject(), 322  
 rejectWith(), 324  
 remove(), 78  
 removeClass(), 62  
 removeData(), 38  
 resolve(), 320  
 resolveWith(), 324  
 serialize(), 44  
 serializeArray(), 44  
 setTimeout, 320  
 split(), 57  
 swing, 206, 211  
 then(), 321  
 tooltip(), 340  
 type(), 41  
 typeof(), 41

unbind(), 87  
 unwrap(), 81  
 wrap(), 80  
 wrapAll(), 82  
 wrapInner(), 81  
 funkcje  
 biblioteki jQuery UI, 206  
 pomocnicze, 39, 256  
 przejścia, 206, 210  
 zwrotne, 118, 314–317, 323

## G

galeria zdjęć, 348  
 generowanie  
 kodu HTML, 71  
 nazw klas, 63  
 przycisku Wstecz, 237  
 grupowanie  
 elementów formularzy, 297  
 elementów zwijanych, 270  
 treści, 153  
 zawartości, 261

## H

HTML, 18  
 HTML5, 25

## I

ikony, 211, 240, 268  
 implementacja  
 obietnicy, 214  
 serwera, 102  
 indeks akapitów, 33  
 informacja  
 o położeniu, 32  
 o postępie prac, 185, 326  
 o wersji, 27  
 instalacja jQuery UI, 123  
 instrukcja return, 30  
 interfejs  
 Promise, 321  
 użytkownika, 121



**J**

JavaScript, 17  
 język  
   biblioteki jQuery, 101  
   CSS, 18  
   HTML, 18, 115  
   HTML5, 25  
   JavaScript, 17  
   Scala, 314  
   serwera, 101  
   XHTML, 25  
   XML, 115  
 jQuery, 18  
 jQuery Core, 202, 208  
 jQuery Mobile, 18, 219, 256  
 jQuery Tools, 20  
 jQuery UI, 18, 123, 208  
 JSON, JavaScript Object Notation, 39, 103  
 JSONP, 118

**K**

kalendarz, 169  
   obsługa zdarzeń, 176  
   określanie opcji, 171  
   określanie wyglądu, 170  
   wywoływanie metod, 177  
 karty, 191  
   obsługa zdarzeń, 194  
   określanie opcji, 193  
   określanie wyglądu, 192  
   wywoływanie metod, 196  
 karty przełączane, 336  
 karuzela, 348  
 klasa  
   accordion-toggle, 348  
   active, 334, 336, 350  
   alert, 343  
   carousel-caption, 350  
   collapse, 348  
   fade, 343  
   in, 343  
   item, 350  
   my-toggle, 209  
   navbar, 331  
   ui-bar, 262  
   ui-bar-b, 262  
   ui-hidden-accessible, 272  
   ui-li-icon, 294  
   ui-resizable, 149, 202  
 klasy CSS, 208, 262  
 klawiatura ekranowa, 275  
 kod serwera, 101  
 kody błędów HTTP, 109  
 kolejka  
   animacji, 215  
   fx, 216  
 kolumny, 249  
 komponent  
   accordion, 270  
   collapsible, 268  
   draggable, 124–129  
     opcje, 126–129  
     przechwytywanie zdarzeń, 129  
     style, 125  
     wywoływanie metod, 130  
   droppable, 130–134  
     opcje, 133  
     przechwytywanie zdarzeń, 132  
     style, 132  
     wywoływanie metod, 134  
   resizable, *Patrz* elementy o zmiennej wielkości  
   selectable, 143–146  
     opcje, 144  
     przechwytywanie zdarzeń, 144  
     style, 144  
     wywoływanie metod, 146  
   sortable, *Patrz* elementy sortowalne  
 komponenty widżetów, 153, 197  
 komunikacja z serwerem, 101  
 komunikaty o wczytywaniu stron, 227  
 konflikt nazw, 26  
 konflikt nazw wtyczek, 308  
 kopiowanie elementów, 73  
 korzeń, root, 93

**L**

liczba akapitów, 35  
 lista  
   jako przełącznik, 279  
   rozwijana, 280, 284  
   wypunktowana, 135, 191  
   zagnieżdżona, 289

**Ł**

łańcuch wywołań, 304  
 łączenie  
   list wypunktowanych, 141  
   obiektów, 41  
   tablic, 41  
   wtyczek i funkcji, 304  
   zbiorów elementów, 47

**M**

manipulacje na drzewie DOM, 73  
 mapy atrybutów, 71  
 menu rozwijane, 332  
 metaznacznik viewport, 220  
 metody  
   elementów o zmiennej wielkości, 151  
   elementów sortowalnych, 140  
   kalendarza, 177  
   komponentów  
     draggable, 131  
     droppable, 134  
     selectable, 146  
   okien dialogowych, 184  
   paska postępu, 187  
   suwaków, 190  
   widżetu  
     akordeonu, 159  
     automatycznego uzupełniania, 164  
     button, 168  
     tabs, 196  
 miniaturka, thumbnail, 293  
 modyfikowanie  
   elementów tablicy, 30  
   stron, 61  
   właściwości elementów HTML, 65

**N**

nagłówki  
   grup, 154  
   HTTP, 112  
 narzędzie  
   Firebug, 254  
   ThemeRoller, 192, 270  
 nawiasy kwadratowe, 32  
 nawigacja, 219  
   pomiędzy stronami, 221  
   po stronie, 191  
 nazwa klasy, 64  
 Node.js, 101

**O**

obiekt  
   Callbacks, 314  
   Deferred, 106, 214, 314, 323  
   document, 25  
   Promise, 106, 319  
   this, 304  
   XmlHttpRequest, 117  
 obiekty obserwowalne, 213  
 obietnica, 106, 213, 320  
 obsługa  
   błędów, 101  
   błędów serwera, 107  
   przekierowań, 110  
   tablic, 41  
   zdarzeń, 23, 85–99  
     AJAX, 116  
     akordeonu, 157  
     elementów o zmiennej wielkości, 150  
     elementów sortowalnych, 136  
     kalendarza, 176  
     komponentów draggable, 129  
     komponentów droppable, 132  
     komponentów selectable, 144  
     okien dialogowych, 181  
     paska postępu, 186  
     przycisków, 167  
     suwaka, 189  
     upuszczania, 131

- urządzeń przenośnych, 233
  - widżetu automatycznego uzupełniania, 161
  - widżetu tabs, 194
  - zadań, 105
  - zadań HTTP, 102
  - odczyt
    - adresów URL, 256
    - danych JSONP, 117
    - danych XML, 115
    - niestandardowych atrybutów, 35
  - odłączanie elementu, 72
  - odnośnik do strony, 221, 224, 228
  - odpowiedź serwera, 111
  - odświeżanie wtyczki, 334
  - odwołanie
    - do biblioteki jQuery, 24, 123
    - do zewnętrznych stron, 228
  - okienka informacyjne, 340
  - okna dialogowe, 178
    - obsługa zdarzeń, 181
    - określanie wyglądu, 179
    - opcje, 180, 181
    - wywoływanie metod, 184
  - okno modalne, 330
  - opcja
    - altField, 171
    - altFormat, 172
    - beforeShow, 176
    - beforeShowDay, 176
    - buttonText, 172
    - closeText, 173
    - currentText, 173
    - duration, 174
    - nextText, 173
    - numberOfMonths, 174
    - showAnim, 175
    - showButtonPanel, 173
    - showOtherMonths, 175
    - stepMonth, 175
    - yearRange, 176
  - opcje
    - elementów
      - o zmiennej wielkości, 149
      - sortowalnych, 136
    - funkcji changePage(), 252
    - kalendarza, 171
    - komponentów
      - draggable, 126
      - droppable, 132
      - selectable, 144
    - okna dialogowego, 180
    - paska postępu, 186
    - przycisków, 167
    - widżetu
      - akordeonu, 156, 157
      - automatycznego uzupełniania, 161
      - kart, 193
      - slider, 189
      - tabs, 193, 194
  - ostrzeżenie użytkownika, 342
- ## P
- para
    - klucz-wartość, 44
    - nazwa-wartość, 39
  - parametry
    - domyślne, 307
    - opcjonalne, 307
  - parametryzacja wtyczek, 306
  - pasek
    - menu, 264
    - narzędzi, 261
    - nawigacyjny, 265–267
    - ostrzeżenia, 263
    - postępu, 185
      - obsługa zdarzeń, 186
      - określanie wyglądu, 186
      - opcje, 186
      - wywoływanie metod, 187
  - pętla for, 28, 29
  - platforma Bootstrap, 20, 329, 353
  - plik
    - 01-app.js, 101
    - bootstrap-button.js, 345
    - bootstrap-carousel.js, 351
    - bootstrap-collapse.js, 348
    - bootstrap-dropdown.js, 334
    - bootstrap-popover.js, 341

- plik
  - bootstrap-tab.js, 338
  - bootstrap-tooltip.js, 341
  - bootstrap-transition.js, 351
  - respond.mini.js, 332
- pliki
  - CSS, 124, 201
  - HTML, 221
- pobieranie
  - danych, 103
  - danych formularza, 43
  - elementów, 32, 57
  - liczb, 187
  - tablic, 31
- podpowiedź, 160, 338
- podział listy na kolumny, 290
- pola
  - formularzy, 271, 275, 279
  - wyboru, 282
  - z atrybutami, 274
- pole
  - do wprowadzania hasła, 277
  - do wyszukiwania, 278
  - tekstowe wielowierszowe, 273
  - typu range, 279
- położenie elementu, 32
- poszukiwanie strony, 108
- powielanie kodu, 327
- procedury obsługi zdarzeń, 93, 129
- przechowywanie danych, 36
- przeciąganie elementów, 124
- przekazywanie
  - funkcji, 63
  - kontekstu, 316, 324, 326
  - nagłówków HTTP, 112
  - obiektu, 27
  - zdarzeń, event delegation, 93, 95
- przekierowania, 110
- przekształcanie
  - akapitu, 178
  - elementów h3, 153
  - elementów w przyciski, 165
  - elementu div, 153, 185, 187
  - listy elementów, 165, 191

- listy w komponent, 135, 146
- obiektu w tablicę, 31
- przełączanie aktywnej karty, 337
- przełącznik, 279
- przesuwanie elementów, 76, 205
- przesyłanie
  - danych z formularza, 114
  - żądań, 116
- przewijanie tekstu, 242
- przycisk Wstecz, 224, 237
- przyciski, 165, 343
  - metody, 168
  - obsługa zdarzeń, 167
  - opcje, 167, 280
  - określanie wyglądu, 166
- przyciski wyświetlające stopkę, 244
- pseudoselektory, 52

## R

- rodzaje pól formularzy, 271
- rozszerzanie obiektów, 41
- rozwijanie zawartości, 346
- rysowanie funkcji przejścia, 210

## S

- schemat kolorów, 247
- selektor hover, 54
- selektory
  - CSS, 26
  - CSS3, 26
  - dynamiczne, 54
  - własne, 57
- separacja
  - elementów, 48
  - kodu HTML i JavaScript, 25
- serializacja danych, 43
- serwer
  - CDN, 25
  - plików, 103
  - WWW, 101
- serwery pamięci podręcznej, 111
- serwery pośredniczące, 111
- skrypt, *Patrz* plik

skrypt wczytujący strony, 253  
 słabe typowanie, 31  
 słowo kluczowe return, 305  
 sortowanie elementów, 138, 141  
 sprawdzanie obiektu jQuery, 27  
 Stack Overflow, 20  
 standard ISO 8601, 172  
 sterowanie
 

- realizacją kodu, 321
- wtyczkami, 308

 stopka, 238–244
 

- grupa przycisków, 240
- położenie, 242
- ukrywanie, 244
- w trybie pełnoekranowym, 246
- wyświetlanie, 239, 241

 stosowanie
 

- obiektów Promise, 321
- okienek informacyjnych, 340
- pól kalendarzy, 273
- rozwijanego menu, 332
- tematów, 247
- wtyczki ScrollSpy, 334

 strony zewnętrzne, 224, 228  
 struktura drzewiasta, 141  
 style
 

- elementów o zmiennej wielkości, 149
- elementów sortowalnych, 136
- komponentów, 201
  - draggable, 125
  - draggable, 132
  - selectable, 144

 sugerowanie wartości, 159  
 suwak, 187
 

- obsługa zdarzeń, 189
- określanie wyglądu, 188
- opcje, 189
- wprowadzanie liczb, 278
- wywoływanie metod, 190

 symulacja akordeonu, 270

## Ś

ścieżka systemowa, 102

## T

tablice
 

- przeglądanie zawartości, 29, 41
- przetwarzanie elementów, 30
- wyszukiwanie elementu, 33

 technologia AJAX, 92, 225–228  
 temat, theme, 247  
 tematy graficzne, 192, 201  
 testowanie
 

- kodu HTML, 36
- możliwości przeglądarek, 44
- typów danych, 39

 treść karuzeli, 350  
 tryb pełnoekranowy, 246  
 tworzenie
 

- animacji, 202
- elementu sortowalnego, 137
- komponentu draggable, 133
- komponentu resizable, 149
- komponentu sortable, 136
- listy wypunktowanej, 71
- łańcucha wywołań, 304
- odnośników do stron zewnętrznych, 221
- parametrów domyślnych, 307
- paska narzędzi, 262
- podpowiedzi, 339
- przełącznika, 280
- przycisków, 165
- rozwijanego menu, 334
- systemu nawigacyjnego, 332
- tablicy, 57
- treści zwijanych, 348
- wielu kolumn, 249
- własnego tematu, 202
- własnych selektorów, 57
- wtyczek, 301
- wtyczek kontekstowych, 303
- wtyczki tworzącej wtyczki, 311

 typy zmiennych, 39  
 tytuł elementu, 223

**U**

- udostępnianie wielu stron, 221
- ukrywanie
  - elementów, 205
  - stopki, 244
  - treści, 62, 268
- umieszczanie treści w karuzeli, 348
- upuszczanie elementów, 130
- urządzenia przenośne, 199, 233
- usuwanie
  - atrybutu, 39
  - danych, 38
  - elementów, 78
  - elementów otaczających, 80
  - klas, 62

**W**

- wczytywanie
  - kodu HTML, 104
  - kodu XML, 115
  - stron, 227, 253
  - zawartości, 253
  - zewnątrznych stron, 224, 226
- wersja biblioteki, 27
- wersje pomocnicze, minor versions, 25
- weryfikacja danych formularza, 113
- węzły drzewa DOM, 255
- widżet, 153
  - accordion, 168, 191
  - akordeonu, 153
  - automatycznego uzupełniania, 159
  - button, 168
  - buttonset, 168
  - datepicker, 169, 173, 177
  - dialog, 181
  - kart, 191
  - okna dialogowego, 180
  - progressbar, 185, 186, 187
  - przycisków, 167
  - slider, 187, 189
  - tabs, 193, 194
- wizualizacja zdarzeń, 87
- właściwości obiektu zwracanego, 259

- właściwość, 65
  - ajax, 45
  - browser, 45
  - context, 59
  - innerHTML, 66
  - maxTransitionWidth, 232
  - opacity, 45
  - support, 45
  - touchOverflowEnabled, 244
  - which, 90
  - xhr.status, 109
- włączanie komunikatów, 227
- wprowadzanie
  - dat, 273
  - liczb, 278
- współczynnik proporcji elementu, 150
- wstawianie elementów, 75
- wtyczka, 299
  - Bootstrap, 350
  - modal, 330
  - Plugin, 309, 311, 313
  - ScrollSpy, 334, 336
  - typeahead, 351
- wtyczki jQuery UI, 123
- wybieranie
  - akapitów, 33, 48
  - daty, 169
  - elementów, 26, 47–60, 282
  - elementów podrzędnych, 49
  - jednego elementu, 280
  - pól formularzy, 52
  - tekstu, 55
  - wybranie elementów, 48
- wygaszanie elementów, 205
- wygląd
  - akordeonu, 154
  - grupy przycisków, 168
  - kalendarza, 170
  - kart, 192
  - komponentów, 125, 201
  - okna dialogowego, 179
  - paska postępu, 186
  - przycisków, 166
  - suwaka, 188
  - widżetu automatycznego uzupełniania, 161

wyłączenie jQuery Mobile, 286  
 wyrażenie \$(this), 98  
 wyróżnianie słowa, 56  
 wyświetlanie
 

- elementów formularzy, 297
- funkcji przejścia, 210
- grup przycisków, 280
- grupy pól, 282
- grupy przycisków, 240
- ikon, 211, 268
- ikon na pasku, 267
- kodu HTML, 105
- komunikatów, 227
- liczby elementów, 292
- list elementów, 288
- listy, 279
- miniaturek, 293
- nagłówków, 112
- nagłówków w listach, 291
- okna modalnego, 330
- ostrzeżenia, 342
- paska ostrzeżenia, 263
- pasków narzędzi, 261
- podpowiedzi, 338
- pól, 277
- pól formularzy, 271, 275
- standardowych pól, 286
- stopki, 239–244
- strony, 220, 223
- treści, 268
- typów zmiennych, 39
- wierszy w pasku, 266
- zdarzeń, 89, 117

 wywołanie
 

- metody, 309
- wtyczki, 322
- zwrotne, callback, 314

## X

XHR, XMLHttpRequest, 117  
 XHTML, 25  
 XML, 115

## Z

zabezpieczanie
 

- obiektu Deferred, 319
- obiektu obietnicy, 320

 zagnieżdżanie
 

- elementów, 80
- grupy akapitów, 82
- list, 289
- selektorów, 53
- zawartości akapitu, 81

 zakres wyboru, 59  
 zapisywanie danych, 36  
 zastępowanie kodu HTML, 66  
 zastosowanie
 

- funkcji fireWith(), 316
- obiektu Callbacks, 314
- przycisków, 344

 zawartość strony, 221, 222  
 zaznaczanie elementów, 143
 

- listy wypunktowanej, 142
- w strukturze drzewiastej, 146

 zdarzenia, 85–99
 

- click, 23, 96
- dragcreate, 129
- jQuery Mobile, 233–236

 zdarzenia związane z
 

- akordeonem, 157
- automatycznym uzupełnianiem, 161
- elementami li, 95
- elementami o zmiennej wielkości, 150
- elementami sortowanymi, 136
- formularzem, 91
- kalendarem, 176
- kartami, 194, 195
- klawiaturą, 89
- komponentami draggable, 129
- komponentami droppable, 130, 132
- komponentami selectable, 144
- myszą, 86
- oknami dialogowymi, 181, 183
- paskiem postępu, 186
- przewijaniem strony, 92
- przyciskami, 167
- suwakiem, 189
- żądaniami asynchronicznymi, 117

zdarzenie

- blur, 92
- change, 92, 265
- document.ready(), 264
- focus, 92
- hidden, 330
- keydown, 89
- keypressed, 89
- keyup, 89
- mousedown, 87
- mouseenter, 87
- mouseleave, 87
- mousemove, 87
- mouseout, 87
- mouseover, 87
- mouseup, 87
- onClose, 177
- onSelect, 177
- pageinit, 264
- ready, 25
- vclick, 237

zestawy kolorów, 270

zewnętrzne pliki JavaScript, 332

zmienianie

- akapitów, 25
- atrybutów, 75
- kodu HTML, 215
- kolejności elementów, 135
- kolorów, 204
- kontekstu wykonania funkcji, 98

- schematu kolorów, 247
- strony z poziomym kodu, 251
- stylów, 344
- tematu, 248
- tytułu elementu, 223
- wielkości elementów, 148
- właściwości atrybutów, 65
- właściwości CSS, 25
- właściwości elementów, 65
- wyglądu komponentów, 201

zmienna \$, 26

znak

- \$, 26, 302
- ?, 119

zwijanie

- grupy, 154
- treści, 346

zwracanie obiektu

- Deferred, 318
- Promise, 319

## Ż

żądania asynchroniczne, 118, 256

żądanie

- DELETE, 103
- GET, 103
- POST, 103
- PUT, 103



# 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>



## Wykorzystaj w pełni potencjał JavaScriptu dzięki bibliotece jQuery!

Czy ktoś pamięta jeszcze czasy, kiedy JavaScript był zimą dla użytkowników? To już przeszłość! Obecnie język ten święci triumfy wraz z HTML5 i CSS3. Dzięki bibliotekom jQuery, jQuery UI i jQuery Mobile wykorzystanie jego potencjału stało się jeszcze łatwiejsze. Teraz możesz błyskawicznie dodawać zaawansowane komponenty i interakcje do Twojej strony, a także tworzyć aplikacje i strony na urządzenia mobilne. Nigdy dotąd nie było to tak proste!

Książka ta jest napisana w sprawdzonej formule receptur i przykładów, dzięki którym błyskawicznie poznasz typowe zastosowania omawianych bibliotek. W trakcie lektury nauczysz się wybierać konkretne elementy na stronie, modyfikować je i usuwać. Ponadto dowiesz się, jak korzystać z modelu zdarzeń, komunikować się z serwerem oraz używać zaawansowanych komponentów interfejsu użytkownika, zawartych w bibliotece jQuery UI. Trzecia część książki została poświęcona jQuery Mobile. Znajdziesz w niej najlepsze sposoby nawigowania, obsługi interakcji oraz dopasowania aplikacji do urządzeń mobilnych. To doskonały przewodnik dla wszystkich twórców stron internetowych oraz aplikacji mobilnych.

### Sprawdź:

- jak łatwo wykorzystać potencjał JavaScriptu
- jak operować na grupach elementów
- jak obsługiwać zdarzenia
- w jaki sposób tworzyć aplikacje na urządzenia mobilne



Addison  
Wesley

**helion.pl**  
księgarnia  
internetowa

Nr katalogowy: 14954



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>



KOD KORZYŚCI

cena: 59,00 zł

ISBN 978-83-246-7703-0



9 788324 677030

Informatyka w najlepszym wydaniu