

PROGRAMMER TO PROGRAMMER™



Wykorzystaj niezwykle możliwości platform
WordPress, Joomla! oraz **Drupal**
i twórz zachwycające witryny dla urządzeń mobilnych!



Programowanie

mobilnych stron internetowych

z wykorzystaniem systemów CMS

James Pearce



Tytuł: Professional Mobile Web Development with WordPress®, Joomla!®, and Drupal®

Tłumaczenie: Tomasz Walczak

ISBN: 978-83-246-3610-5

© 2011 by James Pearce, Palo Alto, California.

All Rights Reserved. This translation published under license with the original publisher John Wiley & Sons, Inc.

Polish edition copyright © 2012 by Helion S.A. All rights reserved.

The Wrox brand trade dress is a trademark of John Wiley & Sons, Inc. in the United States and/or other countries. Used by permission.

Szata graficzna Wrox Brand jest handlowym znakiem towarowym firmy John Wiley & Sons, Inc. w Stanach Zjednoczonych i/lub innych krajach. Wykorzystano na podstawie licencji.

Wiley, the Wiley logo, Wrox, the Wrox logo, Programmer to Programmer, and related trade dress are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. WordPress is a registered trademark of Automattic, Inc. The Joomla! Software and default templates are copyright 2005-2010 Open Source Matters, Inc. Drupal is a registered trademark of Dries Buytaert. All other trademarks are the property of their respective owners. Wiley Publishing, Inc. is not associated with any product or vendor mentioned in the book.

Wiley, logo Wiley, Wrox i logo Wrox, Wrox Programmer to Programmer oraz związane z nimi szaty graficzne są handlowymi znakami towarowymi lub zarejestrowanymi handlowymi znakami towarowymi firmy John Wiley & Sons, Inc. oraz/lub podmiotów z nią związanych, w Stanach Zjednoczonych oraz w ich krajach i nie mogą być wykorzystywane bez pisemnej zgody. WordPress jest zarejestrowanym znakiem towarowym firmy Automattic, Inc. Autorskie prawa własności do kodu źródłowego Joomla! oraz szablonów domyślnych należą do firmy Open Source Matters, Inc. Drupal jest zarejestrowanym znakiem towarowym Driesa Buytaerta. Wszystkie pozostałe znaki towarowe należą do ich właścicieli. Wiley Publishing Inc. nie jest związane z żadnym produktem czy sprzedawcą wymienionymi w tej książce.

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

Wydawnictwo HELION

ul. Kościuszki 1c, 44-100 GLIWICE

tel. 32 231 22 19, 32 230 98 63

e-mail: helion@helion.pl

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

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

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

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

Spis treści

O autorze	13
O redaktorze technicznym	15
Podziękowania	17
Wprowadzenie	19
CZĘŚĆ I ŚWIAT INTERNETU MOBILNEGO	23
Rozdział 1. Wprowadzenie do internetu mobilnego	25
Nieuchronność pojawienia się internetu mobilnego	26
Krótka historia internetu mobilnego	27
Wczesne technologie	27
Usługa i-mode w Japonii	27
Protokół WAP	28
Początki współczesnego internetu mobilnego	29
Nowe medium	30
Powrót do założeń	32
Rozważania na temat internetu mobilnego	34
Rozpoznawanie użytkowników mobilnych	34
Spójność tematyczna	34
Spójność marki	35
Użyteczność przede wszystkim	35
Pamiętaj o mobilności	36
Podsumowanie	36
Rozdział 2. Techniczne omówienie internetu mobilnego	37
Techniczne wyzwania związane z urządzeniami przenośnymi	38
Ograniczenia fizyczne	38
Różnorodność urządzeń	42
Cechy przeglądarek	44
Szybkość i zużycie energii	45
Sieć mobilna	47
Sieci do przesyłu danych	47
Przepustowość i opóźnienie	48
Wprowadzenie do transkodowania	49
Zapory i bezpieczeństwo	51

Inne technologie mobilne	53
Aplikacje i sklepy z aplikacjami	53
Widzety dla urządzeń mobilnych	56
Wiadomości i krótkie kody	57
Kody kreskowe	58
Geolokalizacja i rzeczywistość rozszerzona	59
Podsumowanie	61
Rozdział 3. Jak nadążyć za zmianami?	63
Jak zmieniają się urządzenia?	64
Cechy fizyczne	64
Technologie sieciowe	67
Systemy operacyjne	68
Ewolucja sieci WWW i internetu mobilnego	70
Znaczniki	70
Style	71
Skrypty	73
Osadzane multimedia	74
Klienckie interfejsy API	74
Gdzie znaleźć pomoc?	76
Jednostki standaryzacyjne	76
Społeczności sprzedawców	77
Programy operatorów sieci	78
Niezależne zasoby	78
Podsumowanie	79
Rozdział 4. Najważniejsze przeglądarki mobilne	81
Przeglądarki oparte na silniku WebKit	81
Mobile Safari	82
Android	87
Wersje przeglądarek Nokii	89
Inne wersje	90
Mobile Internet Explorer	91
Opera Mobile i Opera Mini	92
Inne przeglądarki	93
Podsumowanie	94
Rozdział 5. Przybornik programisty rozwiązań mobilnych	95
Wykorzystanie istniejącej witryny	95
Proste techniki statyczne	96
Przenoszenie zarządzanych treści do świata mobilnego	98
Dopracowywanie nowych funkcji mobilnych	102
Tworzenie wersji mobilnej od podstaw	104
Użytkownicy mobilni jako pełnoprawna grupa	104
Współużytkowanie istniejących danych	106

Technologie serwerowe	107
Serwery WWW i świat mobilny	108
Języki i platformy	108
Narzędzia programistyczne	110
Środowiska IDE i edytory kodu	110
Pakiety SDK dla rozwiązań mobilnych i emulatory	112
Narzędzia do testowania	115
Podsumowanie	118
CZĘŚĆ II OGÓLNE TECHNIKI TWORZENIA WITRYN MOBILNYCH	119
Rozdział 6. Struktura witryny mobilnej	121
Struktura witryny i powiązane koncepcje	121
Architektura informacji	121
Punkty wejścia i adresy URL	126
Systemy nawigacji i menu	130
Listy nawigacyjne	130
Ozdabianie menu	133
Ścieżka powrotu	134
Nawigacja w nagłówkach i stopkach	135
Wydeptywanie mobilnych ścieżek	136
Odnosiniki do przełączania wersji	137
Główna zawartość witryny	138
Tekst i czcionka	138
Podział na strony (paginacja)	140
Osadzanie grafiki i multimediiów	141
Formularze	144
Korzystanie z innych funkcji urządzeń	146
Style CSS	147
Style CSS w środowisku mobilnym	147
Optymalizowanie stylów CSS	148
Poziom obsługa JavaScriptu	149
Podsumowanie	151
Rozdział 7. Przełączanie się między przeglądarkami mobilnymi i stacjonarnymi	153
Wykrywanie przeglądarek	153
Sprawdzanie nagłówków	154
Nagłówki User-Agent i transkodery	158
Prosty algorytm wykrywania przeglądarek	161
Wykrywanie przeglądarek na podstawie bazy danych urządzeń	163
Wykrywanie po stronie klienta	165
Przełączanie motywów i rodzaju witryny	167
Wybieranie motywu	173
Zapamiętywanie wyboru użytkownika	174
Korzystanie z domen mobilnych	176
Podsumowanie	177

Rozdział 8. Mobilne interfejsy użytkownika w systemach CMS	179
Rejestrowanie i logowanie	179
Projektowanie formularzy	180
Sprawdzanie poprawności zawartości pól	182
Dopracowywanie wersji mobilnej	184
Usprawnianie logowania	186
Listy materiałów	187
Klawisze dostępu i podział na strony	189
Dekoracje	193
Zwijanie	196
Wyniki wyszukiwania	198
Galerie	200
Wkład ze strony użytkowników	202
Podsumowanie	204
Rozdział 9. Projektowanie pod kątem urządzeń przenośnych	207
Standardowe podejścia	208
Zachowanie wizerunku marki	208
Wykorzystanie natywnych wzorców projektowych	211
Witryna mobilna jako punkt wyjścia	213
Projektowanie interfejsów mobilnych	214
Projekty mobilne oparte na kliencie	216
Wprowadzenie do zapytań Media Query	217
Dostosowujący się projekt	220
Skalowanie rysunków	226
Projekty mobilne oparte na serwerze	228
Uwzględnianie różnorodności	228
Projektowanie pod kątem grup urządzeń	228
Łączenie podejść	232
Podsumowanie	235
Rozdział 10. Szablony i biblioteki mobilne	237
iWebKit	238
Szablony internetowe Nokii	240
jQTouch	241
jQuery Mobile	243
Sencha Touch	246
Podsumowanie	249
CZĘŚĆ III GŁÓWNE SYSTEMY CMS	251
Rozdział 11. WordPress w internecie mobilnym — wprowadzenie	253
Wprowadzenie do systemu WordPress	253
Wpisy, strony i komentarze	254
Multimedia i odnośniki	255
Motywy i widżety	256
Wtyczki	257

dotMobi WordPress Mobile Pack	257
Instalowanie	258
Konfiguracja	261
Konfigurowanie i wzbogacanie motywów	268
Administrowanie witryną w urządzeniach przenośnych	272
WPtouch	273
Instalowanie	274
Motyw wtyczki WPtouch	274
Konfigurowanie	276
WordPress Mobile Edition	280
MobilePress	282
Aplikacja WordPress Mobile	284
Podsumowanie	286
Rozdział 12. System WordPress w internecie mobilnym dla zaawansowanych	287
Rozwijanie własnego motywu mobilnego	287
Nagłówki i stopki	288
Listy wpisów	293
Szczegółowe wyświetlanie wpisów i stron	298
Komentarze	301
Menu i nawigacja	305
Korzystanie z uchwytów i filtrów systemu WordPress	307
Wybieranie motywu	308
Modyfikowanie zawartości	312
Podział na strony	314
Dostosowywanie rysunków	316
Podsumowanie	319
Rozdział 13. Drupal w internecie mobilnym — wprowadzenie	321
Wprowadzenie do Drupala	321
Segmenty i rodzaje zawartości	322
Moduły	322
Blokki	323
Skórki	323
Taksonomia	323
Moduł Mobile Plugin Drupala	324
Instalowanie	324
Konfigurowanie	326
Przegląd interfejsu	334
Moduł Mobile Tools	339
Instalowanie i konfigurowanie	339
Kontrolowanie przekierowań	341
Karta Mobile Roles	342
Moduł Mobile Theme	345
Stosowanie mobilnych skórek Nokii	346
Inne skórki	348
Podsumowanie	350

Rozdział 14. System Drupal w internecie mobilnym — dla zaawansowanych	351
Rozwijanie własnej skórki mobilnej	352
Nagłówki i stopki	356
Segmenty i listy	360
Menu i nawigacja	367
Bloki	368
Komentarze	373
Tworzenie modułów Drupala	380
Wybieranie skórki	380
Modyfikowanie treści	382
Korzystanie z innych modułów	387
CCK	387
Podsumowanie	391
Rozdział 15. Joomla! w internecie mobilnym — wprowadzenie	393
Wprowadzenie do systemu Joomla!	393
Artykuły	393
Sekcje i kategorie	394
Menu	394
Rozszerzenia	394
WAFL	395
Auto Template Switcher	400
Mobilebot	401
Mobile Joomla!	404
TapTheme	410
Podsumowanie	414
Rozdział 16. Joomla! w internecie mobilnym — dla zaawansowanych	415
Rozwijanie szablonu mobilnego	415
Sekcje i kategorie	419
Artykuły	427
Strona główna	429
Moduły i menu	430
Tworzenie dodatku dla systemu Joomla!	432
Wybieranie motywu	434
Przepisywanie zawartości	436
Podsumowanie	438
CZĘŚĆ IV WZBOGACANIE I URUCHAMIANIE WITRYNY	439
Rozdział 17. Frameworki oparte na języku JavaScript	441
jQuery Mobile	442
Lista wpisów	445
Szczegółowy widok wpisów i stron	448
Podsumowanie	450

Rozdział 18. Testowanie i debugowanie witryn mobilnych	451
Stosowanie klientów stacjonarnych	452
Mozilla Firefox	452
Stacjonarne przeglądarki oparte na silniku WebKit	456
Emulatory urządzeń przenośnych	458
iPhone i iPad	458
Android	460
BlackBerry	463
Nokia Series 40 i Symbian^3	465
Palm webOS	465
Opera Mobile	467
Windows Mobile	469
Internetowe laboratoria testowe	471
DeviceAnywhere	472
Perfecto Mobile	473
Zdalny dostęp w Forum Nokia	473
mobiReady	475
Walidatory organizacji W3C	476
Testowanie przy użyciu rzeczywistych urządzeń	477
Podsumowanie	479
Rozdział 19. Ostateczne poprawki	481
Zabezpieczanie witryny	481
Białe listy	482
Unikanie transkodowania za pomocą nagłówków i znaczników	482
Analizowanie ruchu mobilnego	484
Pliki dziennika	484
Analizy dla witryn mobilnych	487
Wyszukiwanie w internecie mobilnym	492
Zarabianie na witrynie	496
Mobilne sieci reklamowe	496
Handel w internecie mobilnym	500
Podsumowanie	502
CZĘŚĆ V ŹRÓDŁA INFORMACJI	503
Dodatek A Dalsza lektura	505
Standardy	505
Języki znaczników w sieci WWW	505
Arkusze stylów	507
JavaScript	507
Sieciowe interfejsy API	508
Najlepsze praktyki	508
Wytyczne od producentów	509

Dodatek B	Przydatne witryny	511
	Przeznaczone dla programistów materiały o systemach CMS	511
	System WordPress	511
	Drupal	512
	Joomla!	512
	Zasoby społeczności związanej z internetem mobilnym	512
	mobiForge	513
	mobiThinking	513
	Wireless Industry Partnership	513
	Mobile Monday	513
	Mobile Web Programming	513
	Quirksmode	514
	mobile-web	514
	wmlprogramming	514
	Społeczności skupione wokół firm	514
	Słowniczek	517
	Skorowidz	523

Projektowanie pod kątem urządzeń przenośnych

W TYM ROZDZIALE:

- Ważne podejścia związane z dostosowywaniem witryny internetowej i marki do odbiorców mobilnych
- Stosowane po stronie klienta techniki pomagające dostosować wygląd witryny do różnych rodzajów urządzeń przenośnych
- Sposoby wykorzystywania kodu z serwera (lub z chmury) do wykonywania podobnych zadań

W tym rozdziale poznasz trendy i techniki z obszaru projektowania witryn na różnorodne urządzenia przenośne. „Projektowanie” zwykle związane jest z wizualnymi odczuciami użytkowników i interakcją z urządzeniem, a nie z projektem architektury całej witryny (ten temat omówiono w innym miejscu).

Najważniejsze zagadnienie, które trzeba poruszyć na początku rozdziału, dotyczy tego, że projektowanie jest wysoce osobistą i twórczą czynnością. Każda witryna jest inna, a w procesie projektowania i implementowania „doświadczeń użytkowników” uczestniczą zarówno projektanci, jak i właściciele witryn oraz marek. Dotyczy to także urządzeń mobilnych, dlatego w tym rozdziale nie opisano jednego sposobu projektowania witryn, a jedynie podano rodzaje zagadnień, które należy omówić i rozważyć w czasie rozwijania eleganckich serwisów mobilnych. Opisywanie estetycznych aspektów grafiki i projektowania interfejsu użytkownika zaledwie wykracza poza główny temat tej książki.

W czasie, gdy powstaje ta książka, można czasem odnieść wrażenie, że w internetowych interfejsach użytkownika w urządzeniach mobilnych pojawia się zaskakująco mało nowatorskich pomysłów. Wiele aplikacji i witryn mobilnych najwyraźniej opartych jest na założeniu, że należy odzwierciedlać pewne standardowe wytyczne (zwłaszcza udostępniane przez producentów urządzeń i przeglądarek), czego efektem jest irytująca jednorodność. Żadne dwie witryny stacjonarne nie wyglądają tak samo, dlatego więc ograniczać się do tworzenia witryn mobilnych podobnych do innych serwisów?

W tym rozdziale poznasz wiele wytycznych i narzędzi. Zobaczysz przede wszystkim, jak zapewnić dostosowanie projektu do różnych urządzeń (zwłaszcza po stronie klienta). Postaraj się jednak wykorzystać to tylko jako punkt wyjścia. Era tworzenia innowacyjnych projektów witryn mobilnych musi się dopiero rozpocząć, dlatego bądź tak twórczy, jak potrafisz!

Każda osoba projektująca interfejs mobilny szybko zauważy, że jednym z największych problemów jest utworzenie interfejsu, który wygląda świetnie w wielu różnych systemach. Możliwość rozwijania witryn oglądanych na urządzeniach przenośnych tylko jednego typu jest rzadkim przywilejem (tworzenie rozwiązań na jedno urządzenie nie jest dobrym pomysłem), dlatego każdy programista i projektant takich serwisów musi znać wiele sztuczek oraz technik radzenia sobie z różnorodnością urządzeń. W tym rozdziale zobaczysz niektóre ważne nowoczesne pomysły pozwalające rozwiązać ten problem.

Standardowe podejścia

W tym podrozdziale poznasz wybrane ogólne pomysły z obszaru projektowania interfejsów użytkownika. Mam nadzieję, że uznasz, iż wiele technik sprowadza się do uwzględniania zdrowego rozsądku, jednak w procesie rozwijania witryn w konkretnym systemie CMS posłużą one jako przydatne wskazówki.

Zachowanie wizerunku marki

Jeśli Twoja obecność w internecie ogranicza się do prywatnego bloga z rzadko pisanymi artykułami, „marka” może obejmować tylko Ciebie, nazwę witryny i domyślny motyw wybrany spośród dostępnych w systemie CMS. Jeżeli jednak jesteś poważnym blogerem, możliwe, że korzystasz z zaawansowanego lub niestandardowego motywu. Ponadto jeśli firma korzysta z systemu CMS do prowadzenia witryny, nazwę, wygląd, styl i ogólny wizerunek marki przekazywany poprzez witrynę należy starannie dostosować do działalności pozainternetowej.

Dlaczego sytuacja miałaby wyglądać inaczej w witrynie mobilnej? Jeśli prowadzisz prosty blog, możesz zastosować standardowy motyw mobilny udostępniany wraz z wtyczką mobilną lub pobrany z sieci WWW. Jednak przy witrynach innego rodzaju urządzenia przenośne są bardzo ważnym kanałem kontaktu z czytelnikami, klientami lub partnerami biznesowymi — czasem nawet istotniejszym niż witryna stacjonarna! Niezwykle ważne jest, aby wygląd stron i wizerunek marki w witrynie mobilnej były zgodne przynajmniej z witryną stacjonarną, a najlepiej z wszystkimi materiałami dostępnymi w internecie i poza nim.

Jeśli poprawnie wykonasz zadanie, przełączanie się między różnymi wersjami serwisów będzie dla użytkowników niezauważalne i zrozumiałe. Przyjrzyj się witrynom sieci ESPN (rysunki 9.1, 9.2 i 9.3). Choć nie wszystkie aspekty (a nawet materiały) witryny stacjonarnej zostały starannie odzwierciedlone w wersjach mobilnych, nie ma wątpliwości, że każda wersja należy do tej samej firmy. Logo, schemat kolorów i wyróżnione informacje są w mniejszym lub większym stopniu spójne, choć układ w każdej wersji jest inny.

Warto porównać te serwisy z blogiem TechCrunch z informacjami z dziedziny technologii. Wersja stacjonarna (rysunek 9.4) rozpoczyna się od paska funkcji w górnej części strony, a dalej znajduje się lista artykułów. W nagłówku widoczne jest duże i rozpoznawalne logo.

ESPN: The Worldwide Leader In Sports

http://espn.go.com/

ESPN: The Worldwide Leader In Sp...

EDITIONS: USA DEPORTES More

CITIES: BOSTON CHICAGO DALLAS LOS ANGELES NEW YORK

Sign In Register

ESPN

LIVE NOW: ESPN3.com, ESPN Texas at Rice

Search

Scores > NCAAFB Full Scoreboard > MLB Tennis (W) Tennis (M) GOLF All Scores >> Regular Season: Week 1 Auto Update: On

my ESPN NFL MLB NBA NHL NCAA FB NCAA BB NASCAR SOCCER MORE SPORTS FANTASY WATCH & LISTEN PAGE 2 & COMMENTARY SHOP & MORE

TOP STORIES TOP VIDEOS: Featuring College Football Highlights

SCORECENTER

PURDUE 3rd 10:20 NOTRE DAME -19

GameCast > Box Score > Conversation >

YDS	Purdue	LEADERS	Notre Dame	YDS
101	R. Marve	PASS	D. Crist	138
27	D. Dierking	RUSH	A. Allen Jr.	49
55	K. Smith	REC	T. Jones	41

GameDay Live

Headlines

UConn at Michigan live on ESPN2/ESPN3. Big Ten blog >
Follow the live game action right here. ESPN3.com >
Get the latest from games around the country. Nation blog >

Transferring data from espn.go.com...

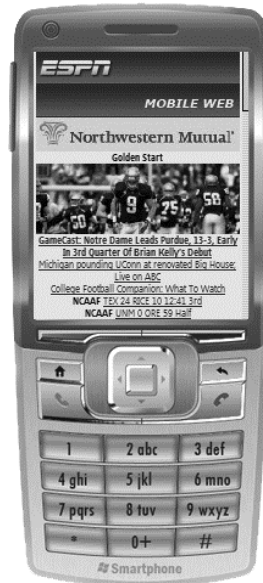
HEADLINES MY HEADLINES ABC News >

- Cardinals decide to release QB Leinart | Blog
- Rams name top pick Bradford as starter | Blog
- No. 4 Florida sloppy in win | No. 9 Iowa rolls
- Kanepe ousts 4th-seeded Jankovic | Pulse LIVE!
- Tiger shoots 65, maintains playoff bid | Scores
- Source: Seahawks let go WR Houshmandzadeh
- Kyle Busch: It's low for Bodine to call me dirty
- A-Rod, Pettitte near return | Yanks' streak at 8
- No. 20 FSU routs Samford in Fisher's debut
- Fantasy: A tool to analyze your draft is here

Rysunek 9.1. Wersja stacjonarna

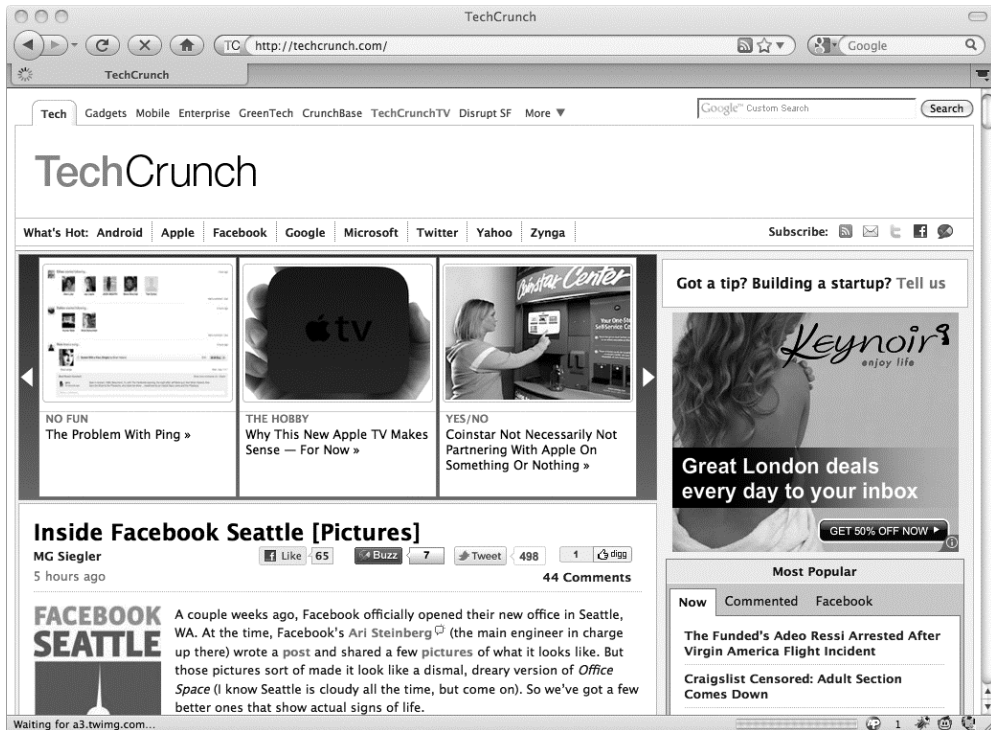


Rysunek 9.2. Wersja mobilna w iPhone'ie



Rysunek 9.3. Wersja mobilna w systemie Windows Mobile

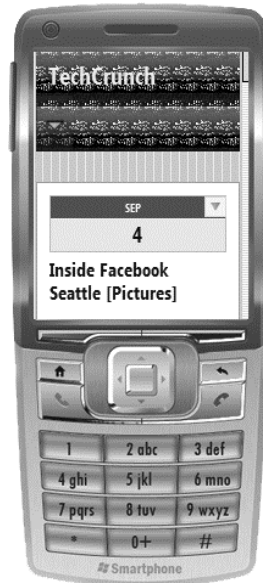
Dziwne jest to, że w czasie powstawania tej książki ani witryna dla urządzeń z wyświetlaczem dotykowym (rysunek 9.5), ani witryna wyświetlana w starszych urządzeniach (rysunek 9.6) nie przypominały wersji stacjonarnej.



Rysunek 9.4. Wersja stacjonarna bloga TechCrunch



Rysunek 9.5. Blog TechCrunch na urządzeniu z wyświetlaczem dotykowym



Rysunek 9.6. Blog TechCrunch na starszym urządzeniu

Dla właściciela witryny lub programisty jest to niewybaczalny błąd, oznaczający utratę okazji do utworzenia spójnego wizerunku marki i jednolitych doświadczeń użytkowników w wielu kanałach.

Należy zapewnić przynajmniej spójność kolorów i zastosować logo z wersji stacjonarnej. Niewłaściwe jest też tworzenie witryn na podstawie natywnego menu iPhone'a (zwłaszcza na urządzenia z systemem Windows Mobile).

Także użytkownik może być zdezorientowany. Czy nadal ogląda witrynę TechCrunch? Dlaczego nie wygląda ona jak wersja stacjonarna? Gdzie jest logo? A może to witryna oszustów podszywających się za twórców serwisu? Dlaczego witryna przypomina menu w iPhone'ie, choć wiem, że korzystam z przeglądarki? Dlaczego witryna wygląda dokładnie tak jak serwis *NFL.com* (rysunek 9.7) i tysiące innych witryn?



Rysunek 9.7. Wersja mobilna przypomina witrynę *NFL.com* i wiele innych serwisów

Czy wynika to z tego, że właściciele wszystkich tych witryn zastosowali domyślny motyw z systemu CMS i nie zastanowili się nad zachowaniem wizerunku marki? Tak, jednak można uznać to za przypadkowy błąd. Internet mobilny jest nowym i nieznanym medium, dlatego brakuje dojrzałości w zakresie sposobów projektowania i rozwijania witryn mobilnych (i prawdopodobnie także umiejętności zapewniania spójnego wyglądu oraz tworzenia projektów równie dobrych jak w stacjonarnych odpowiednikach).

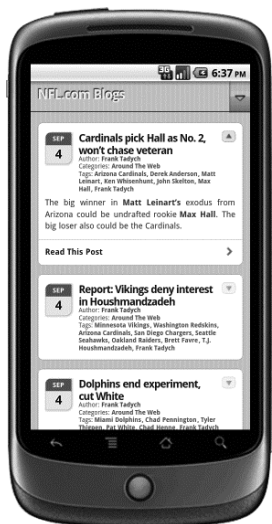
Dobra wiadomość jest taka, że zapewnienie spójnego wyglądu wersji mobilnej i stacjonarnej nie jest specjalnie trudne. Nawet dopasowanie schematu kolorów i zastosowanie wyraźnych elementów wizerunku firmy (na przykład kolorów dodatkowych) robi wielką różnicę, przy czym trzeba zachować ostrożność, aby przy wybranych ustawieniach tekst w wersji mobilnej był wyraźny, czytelny i kontrastowy. Powierzchnia ekranu nie jest duża, dlatego nie trzeba odtwarzać wszystkich dekoracji i aspektów wersji stacjonarnej.

Wykorzystanie natywnych wzorców projektowych

Wraz z pojawieniem się w 2007 roku iPhone'a firmy Apple świat mobilnych interfejsów użytkownika poszedł znacznie do przodu. W iPhone'ie wprowadzono wiele przełomowych zmian w samym urządzeniu, mobilnym systemie operacyjnym i przeglądarce, a wszystko to uzupełniał elegancki i spójny interfejs, dzięki któremu korzystanie z urządzenia było niezwykle proste. iPhone oznaczał początek powszechnego dostępu do internetu mobilnego oraz stanowił punkt odniesienia dla twórców mobilnych interfejsów użytkownika.

Niestety, to, że urządzenie stało się wzorcem, doprowadziło do nieoczekiwanych efektów ubocznych. Ponieważ interfejs użytkownika systemu operacyjnego został tak dobrze przyjęty, początkujący programiści rozwiązań dla internetu mobilnego wykorzystali go jako punkt wyjścia do rozwijania własnych projektów. Sama firma Apple przez udostępnienie wartego przeczytania dokumentu „Human Interface Guidelines for Web Applications” ułatwiła twórcom witryn stosowanie tych samych paradygmatów i wzorców, co w natywnych aplikacjach i samym urządzeniu.

Z jednej strony jest to doskonały punkt wyjścia. Przez zastosowanie technik interfejsu użytkownika znanych odbiorcom z innych części urządzenia można zagwarantować, że korzystanie z witryny będzie łatwe i naturalne. Jednak z drugiej strony łatwo można doprowadzić do skrajności — do sytuacji, w której witryna stanie się *nieodróżnialna* od aplikacji natywnych. Co gorsza, ślepe odwierciedlanie systemu operacyjnego oznacza, że witryna wygląda na dobrze zintegrowaną i spójną w jednym konkretnym urządzeniu, ale zupełnie nieodpowiednio we wszystkich pozostałych. Jak pokazano w poprzednim podrozdziale, akceptowalne (choć niespójne z wizerunkiem marki) jest stosowanie pasków i zaokrąglonych rogów w stylu iPhone’a na blogu, jeśli jest wyświetlany w iPhone’ie, jednak ten sam motyw w Androidzie jest zupełnie nieadekwatny i wygląda dziwnie, co pokazano na rysunku 9.8 na przykładzie wcześniejszej witryny NFL.

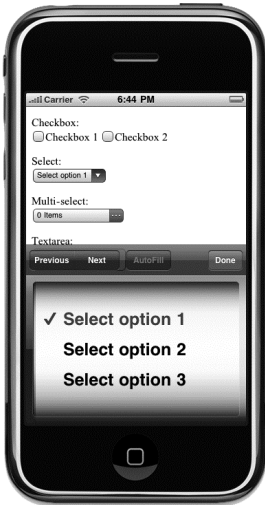


Rysunek 9.8. Interfejs w stylu iPhone’a w systemie Android nie wygląda dobrze

Wyraźnie widać, że złotym środkiem jest połączenie obu podejść. Podstawowe reguły działania interfejsu użytkownika w witrynie nie powinny być obce lub nieoczekiwane dla osoby, która codziennie korzysta z urządzenia do odwiedzania innych witryn. Oczywiście jest też, że witryna nie powinna zakłócać pracy specyficznych dla urządzeń elementów interfejsu użytkownika, na przykład wyjątkowego sposobu wyświetlania znaczników `<select>` w iPhone’ie, co pokazano na rysunku 9.9.

Jednak naiwne odtwarzanie systemu operacyjnego konkretnego urządzenia (w skrajnym stopniu co do piksela) jest prawdopodobnie błędem. Takie podejście zaskakuje użytkowników innych urządzeń, ogranicza kreatywność i, co ważniejsze, uniemożliwia zapewnienie spójnego wizerunku marki względem witryny stacjonarnej.

Warto stosować współczesne i standardowe operacje oraz wzorce interfejsów użytkownika, jednak nie należy robić tego kosztem ograniczania własnych pomysłów z zakresu kosmetyki i estetyki witryny. Przyjrzyj się ponownie witrynie ESPN na iPhone’ie (rysunek 9.2). Autorzy witryny byli wyraźnie



Rysunek 9.9. Specyficzna obsługa znacznika <select> w iPhone'ie

zainspirowani rozwiązaniami z iPhone'a (takimi jak rozwijane menu i przyciski na pasku narzędzi), jednak zbudowali ją w taki sposób, że możliwe było zachowanie marki ESPN i indywidualnego charakteru witryny.

Witryna mobilna jako punkt wyjścia

W społeczności projektantów rozwiązań dla sieci WWW i internetu mobilnego coraz więcej zwolenników zyskuje nowe podejście do projektowania materiałów internetowych. Zgodnie z nim należy zacząć od mobilnej wersji witryny lub aplikacji, a następnie opracować wersję stacjonarną. Podejście to jest atrakcyjne z kilku powodów i zwłaszcza dla twórców witryn lub serwisów, które nie mają istniejącej wersji stacjonarnej. Przykładowo: zwolennicy omawianej techniki podkreślają znaczny wzrost ruchu w internecie mobilnym w ostatnich kilku latach, co oznacza, że niedługo wersja mobilna może generować istotną część łącznego ruchu w witrynach firmy.

Ponadto urządzenia przenośne udostępniają wiele możliwości nieobecnych w urządzeniach stacjonarnych. Zagadnienie to omówiono w rozdziale 5. W urządzeniach z obsługą geolokalizacji, wykrywaniem orientacji, ekranem dotykowym i aparatem funkcje witryny mobilnej mogą być *bogatsze* niż wersji stacjonarnej. W pewnym sensie lepiej jest zbudować wersję mobilną, a później ograniczyć jej funkcje pod kątem użytkowników stacjonarnych. Ta kwestia stanie się ważniejsza w przyszłości, kiedy więcej możliwości urządzenia będzie dostępnych poprzez standardowe interfejsy API dla aplikacji działających w przeglądarkach.

Na zakończenie warto uwzględnić odmienne podejście — trzeba umożliwić wydajne korzystanie z witryny działającej w środowisku mobilnym. Z uwagi na mały obszar fizyczny należy priorytetowo potraktować funkcje usługi lub witryny. Nie ma miejsca na dodatkowe i rozpraszające aspekty projektu lub drugorzędne funkcje, zwłaszcza jeśli utrudniają one użytkownikom szybkie ukończenie podstawowego zadania (które czasem trzeba wykonać w pośpiechu lub w niedogodnym miejscu). Myślenie przede wszystkim o tych użytkownikach pomaga projektantom i programistom skoncentrować się na tym, co sprawia, że aplikacja jest wartościowa.

Podejście „zaczni od wersji mobilnej” może wydawać się nieodpowiednie, jeśli podstawowym celem jest przeniesienie do internetu mobilnego materiałów przechowywanych w systemie CMS. W końcu system już udostępnia świetną witrynę stacjonarną. Jednak każda witryna przechodzi przez etap odświeżania i zmian w projekcie. Kiedy już nauczysz się tworzyć witryny mobilne (a użytkownicy

mobilni zaczęli stanowić duży odsetek wszystkich odwiedzających), bez wątpienia nadejdzie moment, w którym zechcesz postawić ich potrzeby i oczekiwania na równi z potrzebami użytkowników starszej wersji (lub nawet wyżej). Na bardzo praktycznym poziomie uwzględnianie najpierw użytkowników mobilnych jest bardzo przydatne przy projektowaniu interfejsów, które są *stopniowo wzbogacane* i reagują na cechy urządzenia po stronie klienta, co opisano w następnym podrozdziale.

Projektowanie interfejsów mobilnych

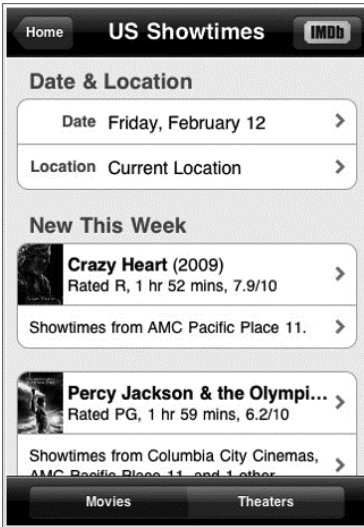
Wraz z pojawieniem się urządzeń przenośnych z ekranami dotykowymi nagle wystąpiło też uzasadnione zainteresowanie interfejsami użytkownika wykraczającymi poza interakcje oparte na myszy i klawiaturze. Autorzy witryny projektowanej i rozwijanej przed wprowadzeniem iPhone'a zakładali — co zupełnie zrozumiałe — że większość użytkowników korzysta z urządzeń wejścia dla komputerów stacjonarnych lub laptopów (klawiatury, myszy albo gładzika), a także dużych wyświetlaczy.

Oczywiście założenia te wpływały na to, w jaki sposób strony, witryny i aplikacje były rozwijane przez około 15 pierwszych lat internetu. Sieć WWW stała się oparta na dokumentach. Strony obejmowały stosunkowo dużą ilość odnośników, a przeglądarki — dodatkowe paski, co pozwalało udostępnić nawigację na całej stronie, odświeżanie itd. Sieć WWW i interfejsy przeglądarek, w których wyświetlano witryny, w niezwykle wysokim stopniu dostosowano do urządzeń wejścia i wyjścia dostępnych dla prawie każdego użytkownika. Odnośniki dostępne w dużej liczbie w zwijanych menu w pasku bocznym są bardzo łatwe do wskazania, przeczytania i kliknięcia, kiedy można posługiwać się precyzyjną myszą.

Jednak w urządzeniach mobilnych każde z tych założeń jest prawdopodobnie błędne. Niedostępna jest pełnowymiarowa klawiatura, którą użytkownik mógłby wykorzystać do wypełniania skomplikowanych formularzy. Ponadto z uwagi na mały ekran okno podglądu jest znacznie mniejsze niż wymiary większości witryn stacjonarnych. Użytkownik może korzystać z klawiszy sterujących fałszywym kursorem „myszy” w przeglądarce, jednak nie działają one tak precyzyjnie ani szybko jak rzeczywista mysz. Bardziej prawdopodobne jest to, że korzysta z urządzenia z ekranem dotykowym, co sprawia, iż w interfejsie użytkownika w ogóle nie istnieje kursor myszy. Przy poruszaniu szerokim palcem po ekranie trudne jest uzyskanie dokładności na poziomie pikseli, a interfejsy oparte na przytrzymywaniu kursora myszy nad elementami nie mają zastosowania.

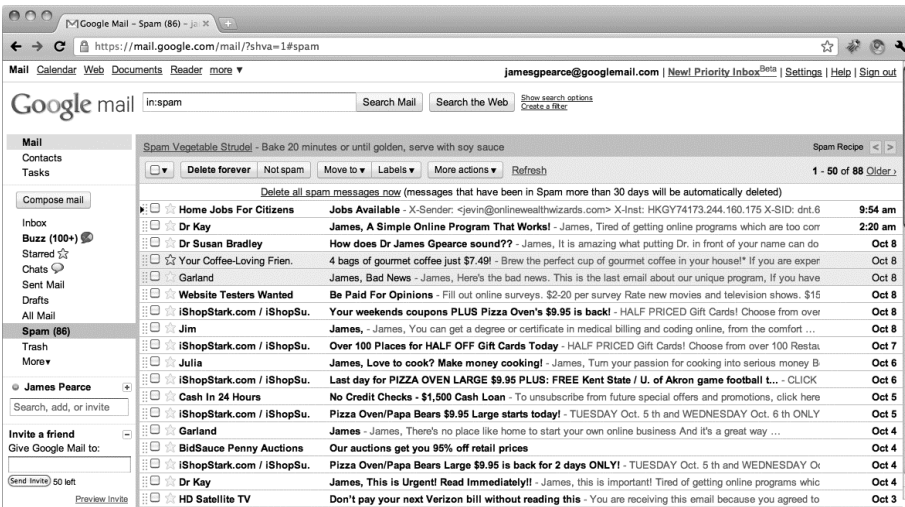
Spółeczność internetowa i programiści stosunkowo powoli zaczęli zwracać uwagę na te drobne różnice oraz wykorzystywać nowe możliwości, jakie dawały urządzenia przenośne. Jednak bardzo duże zainteresowanie natywnymi aplikacjami wynikające z popularności sklepów z aplikacjami na urządzenia Apple'a i dla systemu Android sprawiło, że programiści zaczęli od nowa myśleć o interfejsach użytkownika i interakcji. Pojawiły się nowe i dostosowane wzorce projektowe dla aplikacji natywnych, takie jak specyficzna dla iPhone'a nawigacja przez przewijanie hierarchii, pionowe strony przypominające listy i proste paski narzędzi z przyciskami w dolnej części aplikacji. Na rysunku 9.10 pokazano natywną aplikację Internet Movie Database na iPhone'a, w której zastosowano niektóre z tych technik.

Wszystkie zmiany w standardowych założeniach na temat interfejsów wynikły z tego, że ekrany dotykowe stały się głównym urządzeniem wejścia. Za pomocą palca wskazującego można wygodnie poruszać się w poziomie po hierarchii stron w aplikacji, a przy użyciu kciuka można szybko przesunąć stronę w pionie jednym ruchem. Główne paski narzędzi aplikacji częściej znajdują się w dolnej części ekranu (gdzie można dotrzeć do nich za pomocą kciuka) niż na górnym pasku — tam zwykle widoczny jest tytuł strony i rzadziej używany przycisk *Wstecz*. W środowiskach opartych na ekranie dotykowym wprowadzono też nowe sztuczki z obszaru interfejsu użytkownika (na przykład przeciąganie listy chronologicznej w dół w celu jej odświeżenia). Gesty te często nie mają sensu w świecie zdominowanym przez mysz i klawiaturę.



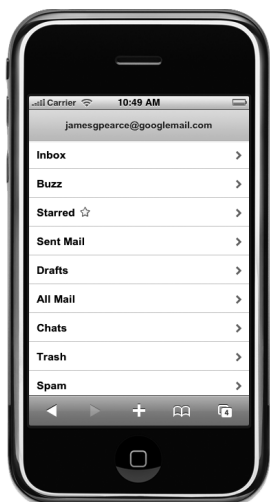
Rysunek 9.10. Natywna aplikacja w iPhone'ie

Dlatego choć pęd do rozwijania natywnych aplikacji klienckich na smartfony początkowo doprowadził do zmniejszenia zainteresowania technologiami opartymi na sieci WWW, pojawiło się w tym czasie wiele innowacji w interfejsach użytkownika. Prawie wszystkie innowacyjne podejścia dotyczą zarówno aplikacji i witryn rozwijanych za pomocą technologii internetowych, jak i rozwiązań tworzonych w językach natywnych. Zastanów się nad aplikacją Google Gmail (rysunek 9.11) i porównaj ją z wersją na iPhone'a (rysunki 9.12 i 9.13). Powinno być oczywiste, jak znacznie zmienił się projekt i interfejs użytkownika aplikacji mobilnych.



Rysunek 9.11. Stacjonarna witryna Google Gmail

Co ważniejsze, aplikacja w iPhone'ie jest wyraźnie oparta na projektach natywnych aplikacji. Podstawowa lista katalogów, która wcześniej znajdowała się w pasku bocznym w stacjonarnej wersji witryny, stała się punktem wejścia w formie listy. Przy wyświetlaniu zawartości konkretnego katalogu główny pasek menu do wykonywania operacji na wiadomościach jest umieszczony wzdłuż dolnej krawędzi ekranu, gdzie można operować kciukiem. Do wybierania (i oznaczania gwiazdkami)



Rysunek 9.12. Witryna Google Gmail w iPhone'ie



Rysunek 9.13. Witryna Google Gmail w iPhone'ie

wiadomości służą duże ikony dostosowane do wielkości palca. Wiele dodatkowych funkcji dostępnych w witrynie stacjonarnej usunięto. Przykładowo: za pomocą interfejsu mobilnego nie można zmienić ustawień dla e-maili lub skonfigurować nowych filtrów.

Jednak choć aplikacja mobilna została opracowana od nowa pod kątem użytkowników mobilnych, to przez zastosowanie tego samego schematu kolorów, spójnych ikon, etykiet i czcionek projektanci nie pozostawili wątpliwości, że nadal jest to serwis Gmail; witryna będzie natychmiast wyglądać znajomo dla użytkowników. Oczywiście nie ma recepty na niezawodne mobilne projekty i konwencje, a w obszarze tym cały czas pojawia się wiele innowacji. Sukces wzorców projektowych w dużym stopniu zależy też od urządzenia używanego do wyświetlania witryny oraz od wyświetlacza i urządzeń wejścia. Wystarczy powiedzieć, że inspirację można czerpać w równym stopniu z udanych natywnych aplikacji mobilnych, jak z innych witryn mobilnych.

Projekty mobilne oparte na kliencie

W społeczności projektantów witryn z dużym zainteresowaniem spotkała się jedna konkretna technika — **dostosowujące się projekty** (ang. *responsive design*). Nazwa pochodzi z artykułu Ethana Marcotte'a (<http://www.alistapart.com/articles/responsive-web-design/>), gdzie autor zasugerował, że serwer powinien zwracać jeden zestaw znaczników i rysunków dla strony, przy czym należy zastosować elastyczne i selektywne style po stronie klienta, aby dostosować układ do różnych wielkości ekranu i urządzeń.

Z jednej strony dla projektanta witryn, który dopiero wkracza w świat internetu mobilnego, podejście to jest bardzo kuszące. Nie trzeba pisać działającego po stronie serwera kodu do zwracania poszczególnym klientom różnych znaczników. Zmiany stylów są uzyskiwane wyłącznie na podstawie zapytań Media Query w stylach CSS. Przeciętny programista witryn zna tę technologię. Z drugiej strony zdaniem niektórych osób stosowanie dostosowujących się projektów to stosunkowo naiwny sposób na udostępnienie aplikacji w internecie mobilnym. Programista nie próbuje ograniczyć ilości kodu ani wielkości rysunków przesyłanych przez sieć do urządzenia przenośnego, a także z pewnością nie zastanawia się nad różnymi funkcjami, które mogą być potrzebne różnym grupom użytkowników.

Jeśli jednak jesteś zdecydowany na opracowanie jednego projektu witryny lub motywu, który działa stosunkowo dobrze w różnych przeglądarkach, warto znać to podejście. Można przynajmniej utworzyć różne projekty dla wersji mobilnej i stacjonarnej, a następnie zastosować dostosowujący się projekt w celu doprecyzowania wyglądu w przeglądarkach z tych dwóch ogólnych grup.

Wprowadzenie do zapytań Media Query

Kluczem do tworzenia dostosowujących się projektów są zapytania Media Query z języka CSS. Są one częścią specyfikacji CSS3 opracowanej przez W3C i umożliwiają projektantom określenie, w jakich warunkach stosowany ma być konkretny arkusz stylów (lub dany zestaw reguł z niego). Urządzenie klienckie (przy założeniu, że obsługuje zapytania Media Query, z czym radzi sobie większość systemów z zaawansowanych smartfonów) przetwarza zestaw prostych reguł, aby ustalić, które arkusze są odpowiednie.

Historycznie sposobem na wybiórcze stosowanie stylów CSS były typy mediów, które są dostępne od dawna i określają różne klasy urządzeń. Poniższy styl deklarowania odnośników w nagłówku dokumentu HTML pozwala odróżnić urządzenia typu screen od handheld.

```
<link rel="stylesheet" type="text/css" href="desktop.css" media="screen" />
<link rel="stylesheet" type="text/css" href="mobile.css" media="handheld" />
<link rel="stylesheet" type="text/css" href="print.css" media="print" />
```

Przeglądarki podające się za stacjonarne pobierają i stosują pierwszy arkusz stylów. Przeglądarki podające się za mobilne wykorzystują drugi arkusz. Niestety, starsze przeglądarki mobilne obsługują typy mediów na różnym poziomie, a współczesne urządzenia, na przykład iPhone, nie podają się w tym kontekście za przenośne. Trzeci arkusz jest stosowany w przeglądarkach stacjonarnych przy drukowaniu zawartości strony.

Wprowadzona niedawno specyfikacja zapytań Media Query umożliwia tworzenie precyzyjniejszych reguł pobierania i stosowania plików arkuszy stylów. Zapytania te są coraz bardziej niezawodnym i precyzyjnym sposobem wybiórczego określania stylów witryn. Oprócz ogólnego typu urządzenia można sprawdzić wiele właściwości sprzętu i wykorzystać je w kodzie warunkowym.

Do właściwości tych należą wielkość ekranu, orientacja i liczba obsługiwanych kolorów. Przykładowo: poniższy kod w nagłówku dokumentu oznacza, że urządzenie powinno wczytać różne arkusze stylów przeznaczone dla trzech przedziałów wielkości fizycznego ekranu.

```
<link rel="stylesheet" type="text/css" href="narrow.css"
  media="(max-device-width:320px)"
/>
<link rel="stylesheet" type="text/css" href="medium.css"
  media="(min-device-width:321px) and (max-device-width:480px)"
/>
<link rel="stylesheet" type="text/css" href="wide.css"
  media="(min-device-width:481px)"
/>
```

Oto właściwości, które można stosować w zapytaniach Media Query:

- **width, height** — wymiary okna podglądu przeglądarki (w pikselach);
- **device-width, device-height** — wymiary fizycznego ekranu (w pikselach);
- **orientation** — orientacja pionowa lub pozioma (określana na podstawie tego, czy wysokość jest większa od szerokości);
- **aspect-ratio, device-aspect-ratio** — stosunek wartości właściwości width do height i device-width do device-height;

- **color, color-index, monochrome** — liczba bitów na składową koloru dla ekranu urządzenia; wielkość tablicy kolorów (jeśli istnieje); liczba bitów na piksel w urządzeniach z ekranem monochromatycznym;
- **resolution** — gęstość pikseli w punktach na cal (dpi) lub punktach na centymetr (dpcm);
- **scan, grid** — proces skanowania stosowany w odbiorniku telewizyjnym; informacja, czy ekran obsługuje tylko wyświetlanie oparte na siatce, takie jak dla czcionek o stałej szerokości lub w terminalach tekstowych.

W kontekście internetu mobilnego prawdopodobnie najczęściej używane są właściwości `device-width`, `device-height` i `orientation`. Można dodać do wymiarów przedrostki `min-` i `max-`, aby określić w zapytaniach wartości graniczne. Warto wspomnieć także o specyficznej dla silnika WebKit właściwości z zapytań Media Query — `-webkit-min-device-pixel-ratio`. Właściwość tę można wykorzystać do zidentyfikowania wyświetlacza Retina o wyższej gęstości, stosowanego w iPhone'ie 4. Z uwagi na zgodność wstecz iPhone 4 skaluje jeden piksel ze stylów CSS na kratkę o wielkości 2 na 2 fizyczne piksele, a wartość wspomnianej właściwości dla tego urządzenia to 2.

W pokazanym przykładzie wykorzystano kilka arkuszy stylów do pokrycia w zasadzie ciągłego przedziału szerokości ekranu. Dlatego naturalne jest, że w ramach tych przedziałów należy zastosować *elastyczny* projekt, w którym elementy płynnie dopasowują się do ekranu (nie występuje tu sztywny układ oparty na siatce). Środkowy przedział obejmuje ekrany o szerokości od 321 do 480 pikseli. Przy sztywnym układzie strony dopasowanym do dolnego krańca tego przedziału na szerszych wyświetlaczach marnowana byłaby nawet jedna trzecia szerokości.

Oto prosty przykład. Poniższy plik jest prawie taki sam jak plik ze znacznikami wygenerowany przez kod do przełączania wersji witryny z rozdziału 7. (typ dokumentu to HTML5).



Do pobrania
w witrynie
helion.pl

```
<!DOCTYPE html>
<html>
<head>
<title>Witaj</title>
</head>
<body>
<h1>Witaj</h1>
<div id="menu">
<a href="index.php">Strona główna</a>
<a href="page1.php">Strona 1.</a>
<a href="page2.php">Strona 2.</a>
</div>
<div id="content">
<p>

</p>
<p>
  Lorem ipsum dolor sit amet...
</p>
<p>
  Praesent sagittis...
</p>
<p>
  Curabitur rhoncus ipsum et...
</p>
<p>
  Donec sodales tristique auctor...
</p>
```

```

</div>
</body>
</html>

```

[Plik responsive/index.html](#)

Przy użyciu elementów `<link>` w nagłówku można dodać arkusze stylów odpowiadające trzem przedziałom szerokości ekranu.

```

<link rel="stylesheet" type="text/css" href="narrow.css"
      media="(max-width:320px)"
/>
<link rel="stylesheet" type="text/css" href="medium.css"
      media="(min-width:321px) and (max-width:480px)"
/>
<link rel="stylesheet" type="text/css" href="wide.css"
      media="(min-width:481px)"
/>

```

Warto zauważyć, że zastosowano tu właściwość `width` zamiast `device-width`. Ta pierwsza dotyczy wielkości okna przeglądarki, a druga — fizycznej wielkości ekranu. Przy testowaniu zapytań Media Query na przeglądarce stacjonarnej przydatna jest właściwość `width`, pozwalająca zobaczyć zmiany stylów przy ręcznej zmianie wielkości okna. Stosowanie jej dla urządzeń przenośnych może początkowo wydawać się ryzykowne — przeglądarka z iPhone'a i inne przeglądarki oparte na oknie podglądu domyślnie przyjmują, że okno podglądu ma szerokość 980 pikseli (co pozwala na poprawną obsługę starszych witryn stacjonarnych), dlatego można podejrzewać, że style dla mniejszych szerokości nigdy nie zostaną aktywowane.

Na szczęście można zmienić wielkość okna podglądu używanego przez przeglądarkę dla danej witryny. Służy do tego właściwość `viewport` w sekcji `<head>` kodu znaczników.

```
<meta name="viewport" content="width=600px" />
```

Można też zastosować poniższą deklarację do ustawienia na stałe okna podglądu na wielkość równą ekranowi fizycznemu.

```

<meta name="viewport" content="
  width=device-width,
  initial-scale=1.0
" />

```

Teraz, przynajmniej w orientacji pionowej, właściwości `width` i `device-width` mają tę samą wartość. Jednak stosowanie właściwości `width` zamiast `device-width` jest poważną zaletą w smartfonach wykrywających orientację ekranu. Po zmianie orientacji właściwość `device-width` przyjmuje nową wartość (480 zamiast 320 pikseli), natomiast fizyczna szerokość (przynajmniej w iPhone'ie, w którym domyślna jest orientacja pionowa) nadal wynosi 320 pikseli. Ponieważ przeważnie pożądaną jest, aby dostosowujący się projekt pasował do szerszego ekranu w orientacji poziomej, można użyć właściwości `width` w zapytaniu Media Query, a ponadto wykorzystać deklarację `viewport`, aby zachować obie omawiane właściwości przeglądarki pod kontrolą. Jeśli chcesz utworzyć *naprawdę* bezpieczne rozwiązanie, możesz posunąć się o krok dalej i wyłączyć możliwość przybliżania i oddalania obrazu przez użytkownika, a także zagwarantować, że przełączanie się między trybem poziomym a pionowym nie prowadzi do zmiany skalowania okna podglądu.

```

<meta name="viewport" content="
  width=device-width,
  initial-scale=1.0,
  minimum-scale=1.0,

```

```

    maximum-scale=1.0,
    user-scalable=false
  " />

```

Dostosowujący się projekt

Wróćmy do arkuszy stylów. Sposób podziału przedziałów wielkości ekranów urządzeń może zależeć od celów projektowych, jednak trzy stosowane tu przedziały pokrywają się mniej więcej z układem pionowym w urządzeniach przenośnych, układem poziomym w urządzeniach przenośnych i ekranami urządzeń stacjonarnych. Teoretycznie plik *narrow.css* obsługuje dowolny ekran o szerokości 320 pikseli lub mniejszej. W praktyce prawdopodobnie będzie przeznaczony tylko dla smartfonów o szerokości 320 pikseli, ponieważ większość zwykłych i uboższych w funkcje telefonów (które zazwyczaj mają węższe ekrany) w ogóle nie obsługuje zapytań Media Query i arkuszy stylów.

Aby witryna wyglądała poprawnie w urządzeniach, które nie obsługują ustawionych warunków, trzeba udostępnić rezerwową arkusz stylów stosowany w każdej sytuacji. Można go podać na pierwszym miejscu listy i umieścić w nim proste style przeznaczone dla docelowych urządzeń przenośnych o mniejszych możliwościach. Jest to także dobre miejsce na zapisanie stylów wspólnych dla wszystkich urządzeń (na przykład schematu kolorów obowiązującego w witrynie).

Następne zagadnienie jest bardzo ważne. Jeśli planujesz stosować zapytania Media Query do uproszczenia obecnej witryny stacjonarnej pod kątem urządzeń przenośnych, spowodujesz problemy we wszystkich urządzeniach, które nie rozumieją takich zapytań. Poniższy kod znaczników jest bardzo złym rozwiązaniem.

```

<link rel="stylesheet" type="text/css" href="huge.css" media="all"/>
<link rel="stylesheet" type="text/css" href="lite.css"
  media="(max-device-width:480px)"
/>

```

Duży (prawdopodobnie stacjonarny) arkusz stylów jest tu wczytywany we wszystkich urządzeniach, a tylko te, które przetwarzają zapytania Media Query (i mają odpowiedni ekran), zastosują drugi arkusz. Jednak w starszych i mniej zaawansowanych urządzeniach dostępny będzie tylko styl *huge.css*, którego bez wątpienia nie zaprojektowano na potrzeby urządzeń z niższej półki.

Oto kolejny przykład myślenia w kategoriach „najpierw wersja mobilna”. Idealne rozwiązanie to utworzenie prostego, ale opartego na stylach i funkcjonalnego interfejsu na potrzeby najmniej zaawansowanych urządzeń oraz zastosowanie zapytań Media Query w celu utworzenia bogatszych, bardziej rozbudowanych interakcji pod kątem urządzeń o szerszych ekranach i potrafiących interpretować wspomniane zapytania. Ta technika to **stopniowe wzbogacanie** (ang. *progressive enhancement*). Programiści witryn stacjonarnych często ją stosują, kiedy muszą poradzić sobie z różnorodnymi możliwościami przeglądarek (choć trzeba przyznać, że nazwa ta często pojawia się w kontekście języka JavaScript, a nie stylów CSS).

Przed stylami specyficznymi dla szerokości ekranu trzeba więc wstawić dodatkowy odnośnik do bezpiecznego, podstawowego arkusza stylów.

```

<link rel="stylesheet" type="text/css" href="basic.css" media="all"/>
<link rel="stylesheet" type="text/css" href="narrow.css"
  media="(max-width:320px)"
/>
<link rel="stylesheet" type="text/css" href="medium.css"
  media="(min-width:321px) and (max-width:480px)"
/>

```



```
<link rel="stylesheet" type="text/css" href="wide.css"
  media="(min-width:481px)"
/>
```

W pliku *basic.css* można umieścić informacje o kolorze, czcionkach i innych podstawowych spójnych aspektach projektu. Na podstawie tego stylu można oczywiście tworzyć dodatkowe arkusze, jednak podstawowe style powinny być niezależne ze względu na prostsze urządzenia, które mogą nie otrzymać dodatkowych stylów.



Do pobrania
w witrynie
helion.pl

```
body {
  background:#ccc;
  font-family:sans-serif;
}

h1 {
  margin:0;
}

#menu {
  padding:10px;
  background:#000;
  color:#fff;
  font-weight:bold;
  font-size:smaller;
}

#menu a {
  color:#fff;
}

#content img {
  width:100%;
}
```

Plik responsive/basic.css

W kodzie nie ma nic szczególnie skomplikowanego, ale to jeszcze nie wszystko. Jest to prosty zestaw stylów, który powinien działać w prawie każdym urządzeniu przenośnym. W starszych urządzeniach z systemem Windows Mobile zastosowane zostaną tylko te style (ponieważ system nie obsługuje zapytań Media Query). Na rysunku 9.14 przedstawiono akceptowalne efekty działania tego rozwiązania.

Przed opracowaniem dalszych stylów warto zobaczyć, jak strona wygląda w przeglądarce w iPhone'ie w orientacji pionowej, co pokazano na rysunku 9.15.

Najoczywistszą wadą jest mała wielkość tekstu (choć rysunek jest rozciągnięty na szerokość ekranu). Wynika to z okna podglądu w iPhone'ie. Widoczna jest strona rozciągnięta do wirtualnego ekranu o szerokości 980 pikseli, a następnie pomniejszona, tak aby pasowała do fizycznego ekranu o szerokości 320 pikseli. Dla rysunku zastosowano styl `width: 100%`, co „wirtualnie” oznacza prawie 980 pikseli na szerokość, dlatego obrazek wygląda na duży, jednak tekst jest tu nieczytelny.

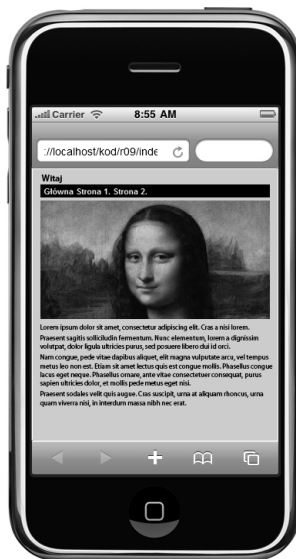
Można rozwiązać ten problem przez ustawienie okna podglądu tak, aby od początku obowiązywało odwzorowanie jeden do jednego względem wielkości ekranu.

```
<meta name="viewport" content="width=device-width" />
```

Nie warto ryzykować podawania dla okna podglądu bezwzględnej szerokości w pikselach (320). Nie zapewnia to zgodności z przyszłymi, coraz większymi ekranami, ani nie pozwala radzić sobie z orientacją poziomą nawet na omawianym urządzeniu. Obecny wygląd tej samej strony pokazano na rysunku 9.16.



Rysunek 9.14. Strona z prostym stylem w urządzeniu z systemem Windows Mobile



Rysunek 9.15. Strona z prostym stylem w iPhone'ie



Rysunek 9.16. Ta sama strona po zmianie stylów

Teraz można dodać ciekawe style do pliku *narrow.css*. Przykładowo: można przyjąć, że urządzenie o niewielkim ekranie, które potrafi przetwarzać zapytania Media Query, ma wyświetlacz dotykowy. Dlatego odnośniki powinny być większe i łatwiejsze do obsługi za pomocą dotyku.



Do pobrania
w witrynie
helion.pl

```
h1 {
  text-align:center;
}
#menu {
  text-align:center;
  background:none;
}
#menu a {
  font-size:medium;
  text-decoration:none;
  padding:5px 10px;
  margin:0;
  border:2px solid #fff;
  background:#000;
}
```

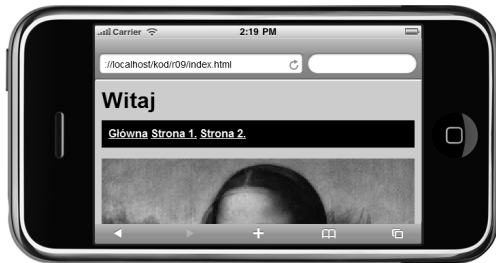
Plik responsive/narrow.css

Tu tytuł jest umieszczony pośrodku okna podglądu o szerokości 320 pikseli, a styl tła usunięto z menu i powiązано z każdym odnośnikiem. Ponadto dodano ramkę, aby odnośniki bardziej przypominały przyciski. Efekt, widoczny na rysunku 9.17, jest spójny ze stylem witryny (trzeba przyznać, że dość zwyczajnym i monochromatycznym), jednak lepiej dopasowany do szerokości ekranu. Nie trzeba wspominać, że można poeksperymentować ze stylami dla silnika WebKit, aby utworzyć bardziej eleganckie przyciski.



Rysunek 9.17. Strona ze stylami dostosowanymi do ekranu dotykowego

Teraz możesz obrócić urządzenie, aby przystąpić do pracy nad następnym stylem, z pliku *medium.css*. Wcześniej jednak warto przyjrzeć się obecnemu wyglądowi strony (rysunek 9.18). Warto zauważyć, że szerokość okna podglądu wzrosła do 480 pikseli (a w zapytaniach Media Query używana jest właściwość *width*), dlatego utracono efekt przycisków z pliku *narrow.css*.



Rysunek 9.18. Strona w orientacji poziomej

Można przenieść menu na lewą stronę ekranu i lepiej wykorzystać szerokość wyświetlacza. 480 pikseli prawdopodobnie oznacza, że używane jest urządzenie z ekranem dotykowym, dlatego ponownie można zastosować style upodabniające odnośniki do przycisków.



Do pobrania
w witrynie
helion.pl

```
h1 {
  border-bottom: 1px solid #fff;
}

#menu {
  position: absolute;
  left: 8px;
  width: 146px;
  padding: 0;
  background: none;
}

#menu a {
  display: block;
  text-align: center;
```

```

font-size:medium;
text-decoration:none;
margin:8px 0;
padding:5px 10px;
border:2px solid #fff;
background:#000;
}

#content {
position:absolute;
left:160px;
padding-left:7px;
border-left:1px solid #fff;
}

#content img {
max-width:304px;
display:block;
}

#content p:first-child {
margin-top:8px;
}

```

Plik responsive/medium.css

Tu menu umieszczono po lewej stronie i ustawiono jego szerokość na 146 pikseli. Odnośniki w menu są elementami blokowymi, dlatego pojawiają się w kolumnie. Odpowiednio dostosowano marginesy, aby odnośniki były odpowiednio oddalone od siebie.

Obszar z treścią (główną część strony) przeniesiono na prawo i określono dla niego taką samą szerokość, jak w orientacji pionowej (przynajmniej na ekranach o szerokości 480 pikseli). Nie jest to konieczne, jednak zapewnia równowagę w projekcie. W rzeczywistości szerokość jest tu nieokreślona, dlatego układ pozostaje elastyczny. Jeśli użytkownik wyświetli stronę na ekranie o szerokości na przykład 400 pikseli, kolumna z treścią zostanie zwężona, a tekst odpowiednio się dopasuje. Efekt pokazano na rysunku 9.19.



Rysunek 9.19. Strona ze stylami dostosowanymi do orientacji poziomej

Jeśli zauważysz, że stosujesz te same style w wielu plikach (tak jak tutaj), powinieneś umieścić kod tych stylów w jednym miejscu, tak aby go nie powielać. Rozbicie stylów na kilka plików powoduje jednak, że urządzenie musi głaszać dodatkowe żądania HTTP.

Jest to dobre miejsce do tego, aby pokazać inny sposób stosowania zapytań Media Query. Wyrażenia można umieszczać *wewnątrz* arkusza stylów za pomocą prostej struktury zagnieżdżonej objętej zapytaniem.

```
@media (min-width:321px) {
  body {
    background:red;
  }
}
```

Za pomocą tej techniki można połączyć arkusze stylów dla wąskich i średnich wyświetlaczy. Wspólne style (na przykład przycisków) przeniesiono na początek arkusza, a układ specyficzny dla ekranów o różnych przedziałach umieszczono w dwóch blokach selektorów pogrupowanych według zapytań Media Query.



Do pobrania
w witrynie
helion.pl

```
#menu {
  background:none;
}

#menu a {
  font-size:medium;
  text-decoration:none;
  padding:5px 10px;
  margin:0;
  border:2px solid #fff;
  background:#000;
}

@media (max-width:320px) {
  h1 {
    text-align:center;
  }

  #menu {
    text-align:center;
  }

  #menu a {
    margin:0;
  }
}

@media (min-width:321px) {
  h1 {
    border-bottom:1px solid #fff;
  }

  #menu {
    position:absolute;
    left:8px;
    width:146px;
    padding:0;
  }

  #menu a {
    display:block;
    text-align:center;
    margin:8px 0;
  }

  #content {
    position:absolute;

```

```

    left:160px;
    padding-left:7px;
    border-left:1px solid #fff;
  }

  #content img {
    max-width:304px;
  }

  #content p:first-child {
    margin-top:8px;
  }
}

```

Plik responsive/narrow-medium-wide.css

Plik można dołączyć do strony przez zastąpienie wcześniejszych deklaracji <link> poniższymi.

```

<link rel="stylesheet" type="text/css" href="basic.css" media="all"/>
<link rel="stylesheet" type="text/css" href="narrow-medium-wide.css"
  media="(min-width:0px)"
/>

```

Dostępne są podstawowy arkusz stylów i połączony arkusz stylów dla dostosowującego się projektu, z warunkiem opartym na zapytaniach Media Query. Choć warunek jest spełniony dla wszystkich urządzeń obsługujących wspomniane zapytania, takie podejście to dobry sposób na wykluczenie pozostałych urządzeń i upewnienie się, że nie próbują zastosować reguł z drugiego arkusza stylów.

Teraz można łatwo dodać do pliku z połączonymi stylami ostatni blok, przeznaczony dla przedziału obejmującego duże ekrany (prawdopodobnie z tabletów lub komputerów stacjonarnych), o szerokości 481 pikseli lub większej. Jedyne różnice to wyświetlanie rysunku wewnątrzkwadratowo, tak aby tekst go opływał, i ograniczenie szerokości elastycznego układu, żeby wiersze tekstu nie były zbyt długie na stacjonarnych monitorach panoramicznych lub na tabletach w orientacji poziomej.

```

@media (min-width:481px) {
  #content {
    max-width:640px;
  }

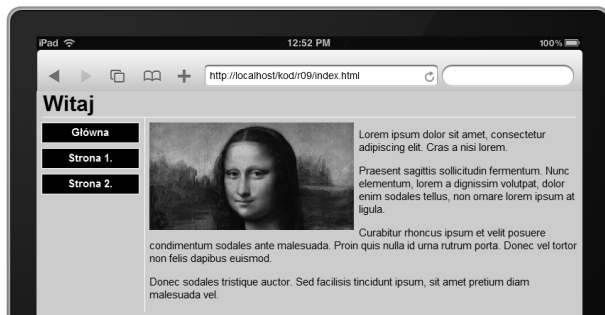
  #content img {
    padding:0 8px 8px 0;
    display:inline;
    float:left;
  }
}

```

Wygląd stylu dla większych wyświetlaczy pokazano na rysunku 9.20. Widać tu iPada w orientacji pionowej (szerokość wynosi 768 pikseli).

Skalowanie rysunków

Ostatnie zagadnienie z obszaru dostosowujących się projektów opartych na stylach CSS dotyczy kwestii, na którą może już zwrócić uwagę. Nie uwzględniono tu wielkości rysunków przesyłanych do urządzeń. Obraz Mona Lizy jest statycznym zasobem, plikiem *lisa.jpg* o wymiarach 304 na 159 pikseli. Zajmuje on skromne 20 kilobajtów, co nie jest problemem przy przesyłaniu danych przez sieć mobilną niezależnie od tego, czy urządzenie przenośne dodatkowo zmniejsza obrazek.



Rysunek 9.20. Strona ze stylem dla urzędzeń o większych ekranach

Łatwo jednak wyobrazić sobie sytuację, w której požądane jest skalowanie większych, bardziej szczegółowych rysunków w dużo większym przedziale wymiarów. Aby rysunek był szczegółowy przy wysokich rozdzielczościach, pierwotny plik musi mieć wymiary przynajmniej równe wymiarom wyświetlacza, co oczywiście oznacza zwiększenie wielkości pliku, który urządzenia o mniejszych ekranach muszą niepotrzebnie pobierać (nie wspomina tu nawet o mocy obliczeniowej potrzebnej do zmiany wielkości rysunku „w locie”).

Pierwszy pomysł może polegać na przygotowaniu zestawu różnych rysunków, które można wybiórczo pobierać na urządzenia. Jednak w tym podrozdziale pokazano, jak dostosować się do różnych urzędzeń po stronie klienta, bez polegania na mechanizmie wykrywania przeglądarek po stronie serwera. Ponieważ używany jest statyczny kod znaczników, nie jest oczywiste, jak wyświetlać rysunki o różnej wielkości i jednocześnie wydajnie korzystać z sieci.

Bardzo atrakcyjną możliwością jest zastosowanie zewnętrznego serwisu do zmieniania wielkości rysunków. `tinySrc`, internetowy serwis tego rodzaju, można wykorzystać do skalowania obrazków na potrzeby dostosowującego się projektu dla urzędzeń przenośnych. Aby użyć interfejsu API serwisu `tinySrc`, wystarczy dodać przedrostek `tinySrc` do adresu URL całego rysunku. Wyobraź sobie, że w kodzie znaczników znajduje się następujący znacznik.

```

```

Wystarczy zastąpić ten znacznik poniższym kodem.

```

```

Serwis `tinySrc` korzysta z bazy danych Device Atlas do zmiany wielkości rysunków w taki sposób, aby pasowały do dowolnego urządzenia zgłaszającego żądanie. Możliwe, że zauważyłeś, iż przy wyświetlaniu obrazka na iPhone'ie wokół strony występuje margines 8 pikseli, dlatego grafika nie powinna mieć szerokości równej dokładnie 320 pikselom (jest to fizyczna szerokość wyświetlacza iPhone'a). Oparty na adresach URL interfejs API serwisu `tinySrc` umożliwia też określenie, że od nowej wielkości rysunku należy odjąć marginesy.

```

```

Kompletna dokumentacja serwisu `tinySrc` znajduje się pod adresem <http://tinysrc.net>.

Projekty mobilne oparte na serwerze

W poprzednim podrozdziale dowiedziałeś się, że ten sam kod znaczników z jednym podstawowym stylem i dodatkowymi stylami opartymi na zapytaniach Media Query pozwala opracować projekt na tyle elastyczny, że będzie działał prawidłowo dla ekranów o różnej szerokości. Warto jednak powtórzyć stwierdzenie z innego miejsca książki — tworząc projekt mobilny, prawdopodobnie należy dążyć do czegoś więcej.

Uwzględnianie różnorodności

Możliwe, że punkt wyjścia (istniejąca witryna stacjonarna lub obecny motyw) jest zbudowany w taki sposób, że nie da się zastosować dostosowującego się projektu. Idealnym rozwiązaniem jest projekt o elastycznej szerokości, jednak od kilku lat znacznie popularniejsze są projekty o stałej szerokości. Trudno wyobrazić sobie przekształcenie szablonu o ścisłych ograniczeniach na wersję odpowiednią dla węższego, mobilnego interfejsu.

Jednak znacznie ważniejsze od tych praktycznych kwestii jest to, że programista poświęcił bardzo mało uwagi bardziej podstawowym zmianom, na które zasługują użytkownicy mobilni. Przesyłając ten sam kod znaczników do zaawansowanego telefonu (jedna skrajność) i 27-calowego stacjonarnego monitora panoramicznego (druga skrajność), programista zakłada, że użytkownicy każdego z tych urządzeń oczekują dokładnie identycznych witryn, usług, architektur informacji itd. W praktyce całościowa strategia rozwijania rozwiązań mobilnych oparta jest na dostarczaniu tego, czego oczekują różne grupy użytkowników. Przecież jedna osoba może w pośpiechu iść chodnikiem, natomiast druga — wygodnie siedzieć przed zamontowanym na ścianie monitorem. Oczywiście jest, że wymagania obu tych osób są inne! W wielu sytuacjach różnice w funkcjach i usługach, które trzeba udostępniać w skrajnie odmiennych urządzeniach i kontekstach, są zbyt duże, aby można je było uwzględnić za pomocą samych stylów CSS i zapytań Media Query. Trzeba wbudować „inteligencję” także po stronie serwera, tak aby mógł wysyłać do poszczególnych urządzeń różne materiały.

Choć w tym krótkim rozdziale nie opisano ponownie przełączania między wersjami stacjonarną i mobilną (temat ten omówiono dość dokładnie w rozdziale 7.), pokazano, co można zrobić w kontekście projektu, jeśli programista *zdecydował się* wykrywać urządzenia po stronie serwera. Skoncentrowano się przy tym na dostosowaniu projektu do różnych grup urządzeń przenośnych.

Projektowanie pod kątem grup urządzeń

We wcześniejszych rozdziałach poruszono technikę podziału docelowych urządzeń na grupy. Przede wszystkim, co oczywiste, należy ustalić, czy urządzenie żądające strony z witryny jest mobilne. Przez zastosowanie wykrywania po stronie serwera, zwłaszcza w połączeniu z bazą danych urządzeń, można stosunkowo łatwo opracować różne interfejsy i wykrywać, jakiego rodzaju urządzenie uzyskuje dostęp do witryny.

Przykładowo: można ocenić różne rodzaje przeglądarek i przygotować dla każdej grupy inną wersję motywu mobilnego. W tabeli 9.1 przedstawiono możliwy podział na grupy.

Urządzenia z tabeli można łatwo podzielić na mniejsze grupy, zwłaszcza jeśli dokładnie wiadomo, które funkcje urządzeń witryna powinna obsługiwać. Przykładowo: jeżeli programista chce w dużym zakresie stosować w witrynie AJAX-a, obsługa odpowiednich technologii będzie równie ważnym kryterium podziału na grupy, co przetwarzanie różnych rodzajów znaczników.

Tabela 9.1. Proste pogrupowanie urządzeń przenośnych

Grupa	Obsługiwane technologie	Przykłady
A	HTML5, CSS3, JavaScript, prawdopodobnie duży obracany ekran dotykowy	Przeglądarki z systemów iOS, Android, webOS, Symbian S60 wersja 5
B	XHTML, CSS2, JavaScript (w ograniczonym zakresie), prawdopodobnie ekran średniej wielkości	BlackBerry wersja 5 i starsze, Windows Mobile wersja 6.5 i starsze
C	XHTML-MP, CSS (w ograniczonym zakresie), brak obsługi JavaScriptu, prawdopodobnie mały ekran	Większość urządzeń wyprodukowanych między 2004 a 2007 rokiem
D	Brak obsługi HTML-a lub XHTML-a, prawdopodobnie mały ekran	Starsze urządzenia z obsługą protokołu WAP sprzed 2004 roku

Na tym etapie warto powiedzieć, że bardzo trudno jest zbudować witrynę działającą w spójny sposób na każdym możliwym urządzeniu i we wszystkich przeglądarkach. Zawsze będą istnieć starsze urządzenia (na przykład grupa D z wcześniejszej tabeli) o ograniczonych możliwościach i coraz mniejszej liczbie użytkowników. Nie warto poświęcać czasu na budowanie lub modyfikowanie witryny pod kątem takich urządzeń. Należy podjąć świadomą decyzję i pominąć projektowanie pod kątem ostatniej grupy oraz zrezygnować z obsługi należących do niej urządzeń.

Niezalecane jest też popadanie w drugą skrajność i obsługiwanie tylko jednej grupy urządzeń. Projekt witryny działający na urządzeniach z tylko jednej konkretnej grupy może być ciekawy i łatwy do opracowania, jednak jeśli nie funkcjonuje poprawnie we wszystkich innych urządzeniach, pozbawiasz się milionów potencjalnych użytkowników, którzy nie korzystają z danego modelu. Może się wydawać, że wszyscy Twoi znajomi ze społeczności programistów witryn używają na przykład iPhone'a firmy Apple (a Ty sam możesz korzystać z jego emulatora na potrzeby tej książki!), jednak z pewnością nie oznacza to, iż wszystkie osoby odwiedzające Twoją witrynę też mają to urządzenie. Użytkownicy nie podziękują Ci za lenistwo i nieuwzględnianie ich potrzeb, co uniemożliwiło dostęp do oferowanych usług. Nieważne, ilu użytkowników mobilnych korzysta z danego modelu telefonu — zawsze będą miliardy innych, którzy używają odmiennych urządzeń.

Wykrywanie, do której grupy należy urządzenie, nie jest nauką ścisłą. Proces ten zależy w dużym stopniu od wybranych kryteriów podziału na grupy. Jednak każda wartościowa baza danych urządzeń (na przykład WURFL lub DeviceAtlas) obejmuje właściwości określające obsługę języków znaczników, a także inne dane, które można wykorzystać do podziału na grupy. Wszystkie te informacje są otrzymywane na podstawie nagłówka user-agent przeglądarki z urządzenia. Przykładowo: jeśli zainstalowałeś bazę DeviceAtlas, poniższy kod pomoże Ci podzielić zgłaszające żądania urządzenia przenośne na cztery grupy, od A do D. E oznacza nieznaną sprzet, a null — urządzenie, które nie jest przenośne.



Do pobrania
w witrynie
helion.pl

```
<?php
```

```
require_once '../lib/da/Api.php';
$tree = Mobi_Mtld_DA_Api::getTreeFromFile(
    "../lib/da_resources/deviceatlas.json"
);
$user_agent = get_http_header('User-agent');

print "Przeglądarka należy do grupy: " . device_group($tree, $user_agent);

function device_group($tree, $user_agent) {
    if (!device_property($tree, $user_agent, 'mobileDevice', 0)) {
        return null;
    }
}
```

```

    }
    if (device_property($tree, $user_agent, 'touchScreen', 0)) {
        return 'A';
    }
    if (device_property($tree, $user_agent, 'markup.xhtmlBasic10', 0)) {
        return 'B';
    }
    if (device_property($tree, $user_agent, 'markup.xhtmlMp10', 0)) {
        return 'C';
    }
    if (device_property($tree, $user_agent, 'markup.wml1', 0)) {
        return 'D';
    }
    return 'E';
}

function device_property($tree, $user_agent, $property, $default) {
    try {
        $value = Mobi_Mtld_DA_Api::getProperty($tree, $user_agent, $property);
        if (is_null($value)) {
            $value = $default;
        }
    } catch (Exception $e) {
        $value = $default;
    }
    return $value;
}

function get_http_header($name, $original_device=true, $default='') {
    if ($original_device) {
        $original = get_http_header("X-Device-$name", false);
        if ($original!='') {
            return $original;
        }
    }
    $key = 'HTTP_' . strtoupper(str_replace('-', '_', $name));
    if (isset($_SERVER[$key])) {
        return $_SERVER[$key];
    }
    return $default;
}

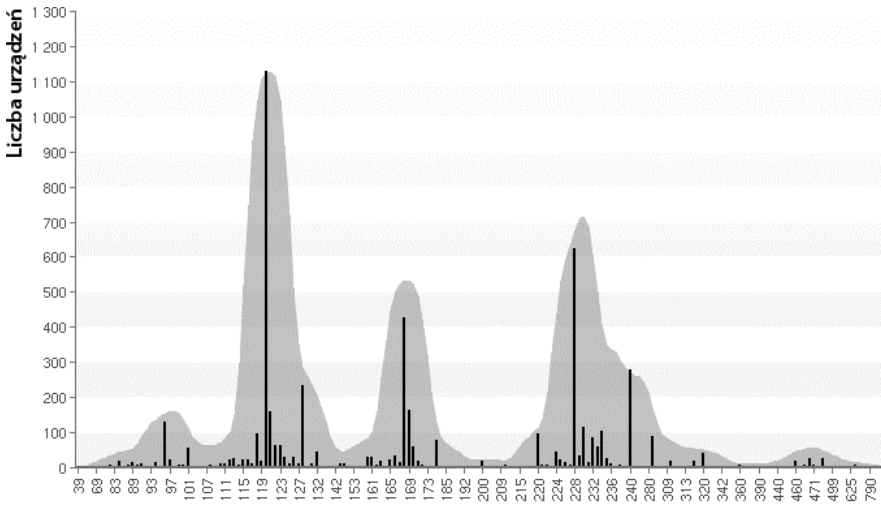
?>

```

Plik da.php

Działanie kodu jest bardzo proste — program wykrywa grupy na podstawie obsługi języków znaczników. Po ustaleniu, do której z głównych grup należy przeglądarka, można wskazać warstwie obsługującej motyw lub style witryny odpowiednie pliki szablonu. Warto zauważyć, że użyto funkcji pomocniczej `device_property` do obsługi wartości domyślnych w sytuacjach, kiedy w bazie danych urządzeń wartość konkretnej właściwości jest nieznaną. Ponownie wykorzystano też roboczą funkcję do sprawdzania nagłówek.

Oprócz grupowania urządzeń na podstawie konkretnych wartości właściwości można też utworzyć kategorie oparte na wartościach ciągłych właściwości, takich jak wielkość ekranu. W rozdziale 2. napisano, że szerokość ekranu urządzenia może wskazywać na rodzaj i ogólne możliwości urządzenia. Na rysunku 9.21 pokazano rozkład liczby urządzeń o określonych szerokościach ekranu.



Wszystkie wartości wykorzystywanej szerokości wyświetlaczy

Rysunek 9.21. Liczba urządzeń o określonych szerokościach ekranu

Jeśli zamierzasz oprzeć się na tym rozkładzie, uzasadnione będzie utworzenie projektów dla ogólnych grup pokazanych w tabeli 9.2.

Tabela 9.2. Grupy urządzeń przenośnych oparte na szerokości ekranu

Grupa	Kryterium	Przykłady
A+	Ponad 400 pikseli	Smartfony (w orientacji poziomej), komunikatory, tablety; prawdopodobnie zaawansowane i z ekranami dotykowymi
A	Od 240 do 400 pikseli	Smartfony (w orientacji pionowej); prawdopodobnie zaawansowane i z ekranami dotykowymi
B+	Od 200 do 240 pikseli	Smartfony; ze średniej i wyższej półki, bez ekranów dotykowych
B	Od 140 do 200 pikseli	Urządzenia ze średniej półki i nowe z niższej
C	Od 110 do 140 pikseli	Starsze urządzenia ze średniej i niższej półki
D	Poniżej 110 pikseli	Starsze urządzenia z obsługą protokołu WAP

Jest mało prawdopodobne, że podział ten idealnie pasuje do wspomnianych wcześniej grup opartych na językach HTML, CSS i JavaScript, może być jednak sensownym przybliżeniem. Przykładowo: poza kilkoma wyjątkami urządzenia o dużej liczbie pikseli na szerokość są nowe i z wyższej półki (a tym samym mają ekran dotykowy). Zaletą stosowania prostej właściwości w rodzaju szerokości ekranu jest szybkość i łatwość oceny urządzenia na podstawie bazy danych w jednej operacji wyszukiwania (uzyskaną wartość można później wykorzystać przy generowaniu stron). Inna korzyść jest taka, że jeśli projektujesz elementy graficzne (na przykład logo i ekrany powitalne) dla różnych grup urządzeń, wiesz, jakie są podstawowe ograniczenia wielkości ekranu w każdej grupie.

Z podziałem na grupy zetkniesz się ponownie w kontekście konkretnych systemów CMS w dalszych rozdziałach. Autorzy kilku wtyczek i modułów dostępnych w systemach WordPress oraz Drupal uwzględnili potrzebę stosowania więcej niż dwóch motywów dla interfejsów mobilnych i umożliwili podział zbioru urządzeń na wiele grup.

Łączenie podejść

Kiedy zastanowisz się nad grupowaniem urządzeń na podstawie ekranu, do głowy może przyjść Ci ciekawy pomysł. Wcześniej w rozdziale opisano stosowanie dostosowującego się projektu jako sposobu na zapewnienie eleganckiego układu witryny na ekranach o różnej wielkości. Jeśli umieścisz urządzenia w odrębnych grupach opartych na ciągłej zmiennej (tak jak w ostatnim podziale), musisz zapewnić obsługę urządzeń z obu skrajnych punktów każdego przedziału. Czy możliwe jest połączenie obu podejść?

Oczywiście odpowiedź brzmi: tak. W inteligentnym podejściu do projektowania pod kątem wielu urządzeń należy harmonijnie stosować techniki działające po stronie serwera i klienta. Choć może to wymagać koordynacji między dwoma wspomnianymi obszarami witryny i procesu jej rozwijania, podejście jest bardzo sprytnie i daje duże możliwości. Pomyśl o tym jak o sposobie na przeniesienie wybranych zapytań Media Query na serwer. Przyjrzyj się poniższemu kodowi, który wykonuje właśnie to zadanie.



Do pobrania
w witrynie
helion.pl

```
<?php
require_once '../lib/da/Api.php';
$tree = Mobi_Mtld_DA_Api::getTreeFromFile(
    "../lib/da_resources/deviceatlas.json"
);

$user_agent = get_http_header('User-agent');

$device_group = device_group($tree, $user_agent);
$touch_screen = device_property($tree, $user_agent, 'touchScreen', 0);

function device_group($tree, $user_agent) {
    if (device_property($tree, $user_agent, 'mobileDevice', 0)) {
        $default_width=160;
    } else {
        $default_width=640;
    }
    $width = device_property(
        $tree, $user_agent, 'displayWidth', $default_width
    );
    if ($width < 241) {
        return 'uproszczona';
    } elseif ($width < 321) {
        return 'wąska';
    } elseif ($width < 481) {
        return 'średnia';
    } else {
        return 'szeroka';
    }
}

...

?><!DOCTYPE html>
<html>
<head>
<title>Witaj</title>
<link rel="stylesheet" type="text/css" href="basic.css" media="all"/>
<link rel="stylesheet" type="text/css" href="<?php
```

```

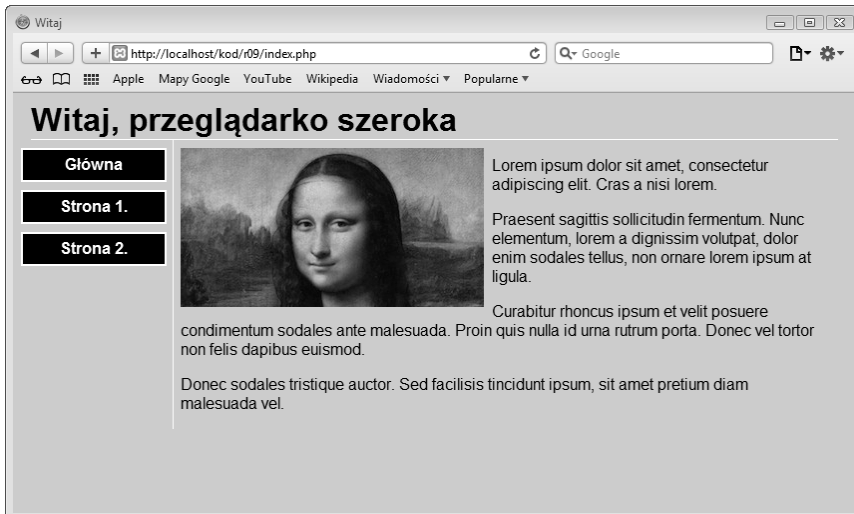
print $device_group;
?>.css" media="all"/>
<?php if ($touch_screen) { ?>
  <meta name="viewport" content="
    width=device-width,
    initial-scale=1.0,
    minimum-scale=1.0,
    maximum-scale=1.0,
    user-scalable=false
  " />
  <?php } ?>
</head>
<body>
  <h1>Witaj, przeglądarko <?php print $device_group;?></h1>
...

```

Plik *responsive2/Index.php*

W tym fragmencie pominięto funkcje `device_property` i `get_http_header`, jednak ich kod pozostał taki sam. Ten program próbuje szybko ustalić, czy urządzenie ma ekran dotykowy i do której grupy określonej według szerokości wyświetlacza należy — szerokiej, średniej, wąskiej lub uproszczonej. Odpowiadają one przedziałom opisanym wcześniej w rozdziale. Warto zauważyć, że domyślna wielkość ekranu jest ustawiana na 160 pikseli dla urządzeń przenośnych i 640 pikseli dla pozostałych urządzeń, co jest przydatne, jeśli precyzyjne wartości są niedostępne w bazie danych. Można wtedy przynajmniej zapewnić płynną degradację.

Zgodnie z oczekiwaniami na rysunkach 9.22 i 9.23 interfejsy w różnych urządzeniach wyglądają inaczej.



Rysunek 9.22. Interfejs dla urządzeń z szerokim ekranem

Kiedy już wiadomo po stronie serwera, do której grupy należy dane urządzenie, można zmienić kod znaczników, co nie było możliwe przy stosowaniu samych zapytań Media Query. W poprzednim przykładzie technikę tę wykorzystano tylko do przełączania arkuszy stylów opartych na grupach (arkusze obejmują rozdzielone fragmenty z wcześniejszego pliku *narrow-medium-wide.css*). Można też usunąć deklarację `viewport` z urządzeń, co do których istnieje podejrzenie, że nie umożliwiają zmiany wielkości (za pomocą dotyku). Oczywiście są to uogólnienia, ponieważ naiwne jest założenie, że



Rysunek 9.23. Interfejs na wąskim wyświetlaczu

wszystkie urządzenia o ekranie o szerokości poniżej 241 pikseli są ograniczone. Jednak podejście to pokazuje, że można zastosować zarówno manipulowanie kodem znaczników, *jak* i dostosowywanie stylów po stronie klienta.

Można wykorzystać tę możliwość i zastanowić się nad różnymi priorytetami poszczególnych grup użytkowników. Jeśli stworzysz witrynę firmy, sklepu lub restauracji, możesz bardziej wyeksponować dane kontaktowe i mapę dojazdu w wersji mobilnej niż stacjonarnej (gdzie dane kontaktowe można umieścić w dolnej części strony).



Do pobrania
w witrynie
helion.pl

```
...
<body>
<?php if (device_property($tree, $user_agent, 'mobileDevice', 0)) { ?>
  <div class='call_to_action'>
    Zadzwoń do nas na numer <a href='tel:555-1234-567'>555-1234-567</a>!
  </div>
<?php } ?>
<h1>Witaj, przeglądarko <?php print $device_group;?></h1>
...
```

Plik index.php

Zmiany tego rodzaju są dużo łatwiejsze do zaimplementowania, jeśli można zmodyfikować kod znaczników, zamiast tylko określać układ za pomocą wybiórczo stosowanych stylów CSS. Już samo to jest istotnym argumentem przeciw stosowaniu projektów opartych tylko na zapytaniach Media Query i stylach CSS.

Co ciekawe, reagowanie po stronie serwera zapewnia kilka dodatkowych możliwości w zakresie odpowiedzialnego udostępniania rysunków na urządzenia przenośne. Zamiast wysyłać ten sam rysunek na urządzenia w celu dostosowania jego wielkości do dowolnego rozmiaru ekranu, można wstępnie przygotować obrazki o odpowiedniej wielkości dla urządzeń z każdego przedziału.

```

```

Przy stosowaniu tego kodu trzeba udostępnić statyczne pliki o nazwach *lisa_narrow.jpg*, *lisa_medium.jpg* i tak dalej, jednak w ten sposób można utworzyć rysunki o mniej więcej odpowiedniej wielkości i uniknąć sytuacji, w której trzeba pobierać bardzo duże obrazki i zmniejszać je do małych

rozmiarów, co jest kosztowne. Inna możliwość to automatyczne zmienianie wielkości po stronie serwera, tak aby rysunki były idealnie dostosowane do każdego żądania, choć to rozwiązanie wymaga nieco więcej kodu. Jak zobaczysz, wiele wtyczek do systemów CMS udostępnia podobną funkcję.

Podsumowanie

Rozdział ten był stosunkowo teoretyczny, po części dlatego, że dziedzina projektów mobilnych dopiero się rozwija, ale też z uwagi na to, iż opracowanie dobrego projektu jest bardzo indywidualnym procesem i trudno dokładnie określić, które techniki będą odpowiednie przy uwzględnieniu Twoich umiejętności, systemu i witryny. Poznałeś jednak pewne podstawowe zasady projektowania interfejsów dostosowywanych zarówno po stronie serwera, jak i po stronie klienta. Dowiedziałeś się też, że warto próbować radzić sobie z różnorodnością urządzeń na wiele sposobów. Jeśli chodzi o estetyczne inspiracje w tym szybko rozwijającym się obszarze, możesz odwiedzić witrynę Mobile Awesomeness (www.mobileawesomeness.com), gdzie dostępne są galerie współczesnych projektów dla internetu mobilnego.

W następnym rozdziale poznasz szereg szablonów i bibliotek, które można wykorzystać do przyspieszenia podróży po świecie projektów mobilnych. Następnie przejdziesz do udostępniania materiałów z istniejących systemów CMS w internecie mobilnym.

Skorowidz

A

ADC, Apple Developer Community, 77
administrator, 99
administrowanie witryną, 273
AdMob, 497
akcje, 257
algorytm PageRank, 493
algorytm wykrywania przegładek, 310
analiza danych internetowych, 268
analizowanie ruchu mobilnego, 484
 Google Analytics, 489
 PercentMobile, 487
 pliki dziennika, 484
Android, 69, 461
 okno podglądu, 89
Android Market, 54
Android SDK, 113
Android SDK Manager, 462
APC, 164
aplikacja
 Active Sync, 472
 DeviceAnywhere Studio, 474
 iOS Simulator, 460
 konsola diagnostyczna, 460
 menu Hardware, 460
 opcja Device, 460
 panel Settings, 460
 Microsoft Virtual PC 2007, 471
 Nokia Maps, 60
 WordPress Mobile, 285
aplikacje
 internetowe, 74
 mobilne, 215
 natywne, 215
Apple XCode, 113
arkusze stylów, 71
artykuł, 393
artykuł po zastosowaniu stylów, 428
artykuły, 427

artykuły w formie bloga, 426
artykuły w tabeli, 426
Auto Template Switcher, 400
automatyczne uzupełnianie, 200
automatyczny podział na strony, 336
AVD, Android Virtual Device, 461

B

baner reklamowy, 410
baza danych, 98
baza danych DeviceAtlas, 177, 227
baza danych WURFL, 163, 177, 332
bezpieczeństwo, 51
białe listy, 482
biblioteka
 Ext JS, 151
 iWebKit, 172
 jQuery, 241, 488
 jQuery Mobile, 243, 442, 446
 lista wpisów, 445
 styl formularzy, 449
 widok wpisów i stron, 448
 jQuery UI, 243
 PercentMobile, 488
 switcher, 175
BlackBerry, 464
BlackBerry OS, 69
blog TechCrunch, 210
blok Switch to and from mobile version, 331
bloki, 368
BOM, Browser Object Model, 166
BONDI, 75

C

CAPTCHA, 186
CDMA, 47
CDMA2000, 47

CHTML, 405
C-HTML, 27
crawlers, 486
crawlers mobilne, 492
CSS, Cascading Style Sheets, 71
CSS3, 72, 77
curl, 454
czas nawiązywania połączenia, 48
czas powrotu pakietu do nadawcy, 48
czas przesyłania danych, 48
czcionka, 139
czcionka bezszeryfowa, 139
czcionka szeryfowa, 139

D

debuger Dragonfly, *Patrz* Dragonfly 468
debugowanie, 451
debugowanie modelu DOM, 468
debugowanie zdalne, 468
deklaracja viewport, 219
dekorowanie list, 195
DeviceAnywhere, 115, 473
DeviceAtlas, 164
dodanie stylów, 378
dodawanie

- bloków, 370
- komentarzy, 301
- nowych pól, 388
- reklam mobilnych, 499

dokument

- Guidelines for Web Content Transformation Proxies, 482
- UAProf, 156
- W3C Mobile Web Best Practice, 195

dokumentacja WordPress Codex, 294
domena mobilna, 176
domyślne ustawienia marginesów, 203
dostosowujące się projekty, responsive design, 216
dostosowujący się projekt, 220
dostosowywanie rysunków, 316
dotMobi WordPress Mobile Pack, 257

- dostosowywanie motywów, 269
- instalowanie, 258
- konfiguracja, 261
- pobieranie, 258
- przełącznik, 261
- przetwarzanie danych, 268
- widzety, 263

Dragonfly, 468

- debugowanie zdalne, 469
- podłączanie emulatora, 470
- strona z przeglądarki mobilnej, 470

Drupal, 321

- bloki, 323, 368
- CCK, 387
- interfejs, 334
- komentarze, 373
- menu, 367
- menu domyślne, 337
- menu Navigation, 336
- menu Primary links, 336
- menu Secondary links, 336
- Mobile Plugin, 324
- Mobile Theme, 345
- Mobile Tools, 339
- moduły, 322
 - agregator, 322
 - blog, 322
 - book, 322
 - comment, 322
 - contact, 323
 - field, 323
 - forum, 323
 - menu, 323
 - search, 323
 - taxonomy, 323
- nagłówek i stopka, 356
- nawigacja, 367
- rodzaje zawartości, 322
- segment, 322, 360
- skórka mobilna, 338, 352
- skórki, 323
- taksonomia, 323
- tworzenie modułu, 380
- wybieranie skórki, 380

Drupal 7, 322
drzewo decyzyjne, 154
dynamiczne generowanie stron, 179

E

Eclipse, 78, 461
edytor kodu, 110
ekran dotykowy, 41, 140
ekran laptopa, 140
elastyczny projekt, 218
element nawigacyjny, 133
EMS, Enhanced Messaging Service, 57

emulator, 112
 Androida, 464
 BlackBerry, 465
 funkcja rejestrowania skryptów, 465
 firmy Microsoft, 114
 firmy Nokia, 114
 iPad, 459
 iPhone 4, 459
 Opera Mobile, 468, 469
 Palma, 114
 Palm Pre, 467
 systemu Symbian, 467
 systemu Windows Mobile, 471

emulatory
 smartfonów, 458
 urządzeń, 78, 458

F

filtry, 257
 filtry rodzajów reklam, 498

Firebug, 452
 debugowanie, 454
 obsługa stylów, 455
 testowanie, 455

Firefox
 Live HTTP Headers, 456
 menu Narzędzia, 452
 przełącznik User-Agent Switcher, 452
 wtyczka Firebug, 452

firma
 Adobe, 74
 Aduity, 283
 Apple, 30, 53
 AT&T, 27, 49
 Automattic, 253, 286
 BraveNewCode Inc, 274
 DeviceAnywhere, 473
 DoCoMo, 28
 dotMobi, 117
 Google, 69
 Nokia, 474
 Openwave, 159
 Opera, 468
 Paca Mobile Center, 480
 Palm, 69, 466
 PayPal, 500
 PercentMobile, 268
 Perfecto Mobile, 474
 Qualcomm, 53

Research In Motion, 464
 RIM, 69, 91, 114
 Siruna, 395
 Unwired Planet, 27

Flash Lite, 144

format animowanych rysunków, 159

format QR Code, 58

formularz
 logowania, 181, 186, 335
 na komentarz, 205
 po zastosowaniu stylów, 373
 rejestracyjny, 180
 w iPhone'ie, 145, 183
 w systemie Android, 145
 w systemie Nokia Series 40, 145
 w systemie Windows Mobile, 145, 183

formularze, 144, 240

ForumNokia, 78

framework Sencha Touch, 246

framework webowy, 109

funkcja
 add_query_arg(), 384
 bloginfo(), 445
 get_http_header, 161, 173
 getTemplate, 401
 onReady, 248
 request_deserves_jqm(), 443
 request_is_mobile, 173
 skorka_mobilna_links(), 364
 the_post(), 295
 theme(), 169
 theme_links(), 367
 theme_textarea(), 375
 theme_textfield(), 375
 wp_list_comments(), 302

funkcje
 do obsługi wyrażeń regularnych, 307
 do rejestrowania skryptów, 465
 do wyodrębniania nagłówków, 161
 mobilne, 102

G

galerie, 201

geolokalizacja, 75

gesty wielodotykowe, 463

Google AdSense, 499

Google Analytics, 489
 mapa źródeł ruchu, 490
 raport, 491

Google Checkout, 501
GPS, Global Positioning System, 59
grupa Smartphone, 407
grupowanie urządzeń przenośnych, 229
grupy urządzeń, 231
grupy znaczników i komponentów, 240
GSM, Global System for Mobile Communications, 47

H

handel elektroniczny
 Google Checkout, 501
 PayPal, 500
handel w internecie mobilnym, 500
HDML, 27
hierarchia nadmiernie szeroka, 125
hierarchia witryny, 122
HSDPA, 48
HTML5, 31, 71
HTTPS, 52

I

IDE, Integrated Development Environment, 111
ikona wpisu, 255, 277
importowanie listy nagłówków, 453
instalowanie Mobile Plugin, 324
interfejs API, 74
interfejs API motywów, 256
interfejs FastCGI, 108
interfejs użytkownika, 212
interfejsy w różnych urządzeniach, 233
interfejsy wyszukiwarek, 493
internet mobilny, 25, 37
iWebKit, 238

J

J2ME, 53
jakość stron mobilnych, 116
język
 (X)HTML, 108
 ASP.NET MVC, 109
 C#, 108
 CSS, 217
 CSS2, 147
 Django, 109
 HTML5, 108
 JavaScript, 73

 natywny, 215
 PHP, 108
 Python, 108
 Ruby, 108, 109
 VB.NET, 108
 WML, 155
 XHTML-MP, 108
 znaczników, 27, 70, 159

Joomla!, 393

 artykuł, 394, 427
 kategoria, 394, 419
 menu, 394, 432
 rozszerzenia, 394
 dodatki, 395
 języki, 395
 komponenty, 394
 moduły, 394
 szablony, 394
 rozszerzenie
 Auto Template Switcher, 400
 Mobile Joomla!, 405
 Mobilebot, 402
 WAFL, 395
 sekcja, 394, 419
 strona główna, 429
 TapTheme, 411
 tworzenie dodatku, 432
 tworzenie szablonów, 416
 włączanie dodatku, 434
 wybieranie motywu, 434
 znaczniki, 417

K

karta

 Device Groups, 332
 Mobile Roles, 342
kaskadowe arkusze stylów, 71
katalog
 com_content/article, 427
 da, 164
 da_resources, 164
 images, 444
 JOOMLA/plugins/system, 433
 motywów, 269
 szablonu, 416
 wtyczka_mobilna, 308
 wurfl, 163
 wurfl_resources, 163
kategoria, 394

klawiatura QWERTY, 41
 klawisze dostępu, 190
 klient testowy firmy Perfecto Mobile, 475
 kod

- działający po stronie klienta, 248
- HTML strony, 454
- kreskowy, 58, 502
- śledzący, 488
- znaczników, 244

komentarze, 204

komponenty jQuery Mobile

- formularze, 246
- listy, 246
- paski narzędzi, 246
- przyciski, 246
- strony i hierarchie, 246
- style dla treści, 246

konfiguracja

- dotMobi WordPress Mobile Pack, 261
- emulatora systemu webOS, 467
- Mobile Joomla!, 406
- Mobile Plugin, 326
- Mobile Tools, 340
- Mobilebot, 403
- Smartphone, 407
- TapTheme, 412
- WordPress Mobile Edition, 282
- WPtouch, 277

konfigurowanie

- elementu menu, 413
- menu, 414
- profilu, 366
- serwerów transkodujących, 482

kontrolki paginacji, 193

korporacja TigTags, 487

krótkie kody, 57

L

lista, 187, 189, 199, 240

- artykułów z sekcji, 420
- materiałów, 187
- modułów, 325, 398
- nawigacyjna, 130
- nieuporządkowana, 187
- sekcji, 420
- wpisów, 293, 445
- wpisów ze stylami, 297
- z ikonami, 196
- z metadanymi, 191

Live HTTP Headers, 456

logowanie, 179

LTE, Advanced Long Term Evolution, 48

Ł

łącze dosyłowe, 49

M

marginesy i czcionki, 292

maszyna wirtualna, 467

m-commerce, 500

mechanizm wyszukiwania, 199

menu, 305

menu domyślne, 305

menu po zastosowaniu stylów, 306

menu rozwijane, 415

menu w obszarze mobile, 414

metoda

- getElementByClassName, 198
- getElementByTagName, 198

midlet, 53

MMS, Multimedia Messaging Service, 57

MNG, 159

mobiForge, 257

Mobile Analytics, 268

Mobile Joomla!, 405

Mobile Pack, 257

Mobile Plugin, 324

- instalowanie, 324
- konfiguracja, 325
- zmiana wielkości rysunku, 338

Mobile Safari, 82

- okno podglądu, 85

- przycisk Add to Home Screen, 84

Mobile Switcher, 269

Mobile Theme, 345

Mobile Tools, 339

- instalacja, 339
- karta Mobile Roles, 342
- konfiguracja, 339
- przekierowania, 341
- przełączanie skórek, 341
- włączanie modułu, 339

Mobile Web Initiative, 77

Mobilebot, 402

MobilePress, 283

- motyw, 285

- opcje, 284

mobilne sieci reklamowe, 496
 mobiReady, 476
 model DOM, 245, 455
 moduł

- Auto Template Switcher, 401
- CCK, 387
- Content Construction Kit, 387
- Fields, 387
- Mobile Plugin, 324
- Mobile Theme, 345
- Mobile Tools, 339
- mod_autotemplateswitcher, 400
- wurfl, 332

 moduły, 322
 modyfikowanie zawartości, 312
 motyw

- mobilny, 260, 287
- Purity, 429
- w systemie WordPress, 271
- wtyczki Mobile Edition, 282
- wtyczki WPTouch, 275

 motywy, 99
 Mozilla Firefox, 452
 MSDN, 78
 MSIE, Microsoft Internet Explorer, 401
 MVC, Model-View-Controller, 109
 MVT, Model-View-Template, 109

N

nadawanie stylów

- formularze, 240
- listy, 240
- nawigacja, 240
- tabele, 240
- treść, 240

 nagłówek

- unikanie transkodowania, 482
- Accept, 155, 159, 161
- Cache-Control, 483
- NN-Profile, 156
- Profile, 156, 161
- Referer, 175, 486
- User-Agent, 155, 158, 486
- Vary, 483
- Wap-Profile, 156
- X-Wap-Profile, 156, 161

 nagłówki

- HTTP, 157
- i stopki, 288, 291, 356

MIME, 108

- z rysunkami, 366

 nagrywanie dźwięku, 147
 narzędzia do testowania, 115
 narzędzia programistyczne, 110
 narzędzie

- Analog, 484
- Android SDK Manager, 461
- Google Wireless Transcoder, 495
- mobileOK Checker, 115

 nawigacja, 240, 305

- główna, 134
- pomocnicza, 136
- w nagłówkach, 135
- w stopkach, 135

 NFC, Near Field Communication, 67
 nieprawidłowy certyfikat, 52
 Nokia Mobile Themes, 346

O

obiekt navigator, 166
 obsługa JavaScriptu, 149
 obsługa płatności, 501
 odnośnik manage fields, 387
 odnośnik readmore, 425
 odnośniki, 131, 137, 256
 odnośniki do kategorii, 365
 odnośniki do metadanych, 191
 okno podglądu, 219
 OMTP, Open Mobile Terminal Platform, 75, 77
 opcja

- Component on home page, 410
- debugowania zdalnego, 469
- Mobile Analytics, 488
- Show pathway, 410
- template switching, 396

 opcje

- rozszerzenia WAFL, 396
- skórki, 357
- wtyczki MobilePress, 284

 opis języka HTML5, 77
 opóźnienie, 48
 optymalizowanie stylów CSS, 148
 organizacja

- dotMobi, 257
- W3C, World Wide Web Consortium, 72, 115, 477

 orientacja ekranu, 219

osadzane multimedia, 74
 osadzanie grafiki, 141
 osadzanie multimediów, 143

P

paginacja, 140

pakiet

- Android SDK, 113, 461
- Apple XCode, 113
- iWebKit, 239
- narzędzi Xcode, 459
- Palm SDK, 466
- SDK, 112
- szablonów TapTheme, 411
- Visual Studio, 471

Palm webOS, 466

panel administracyjny, 273

panel Device Groups, 334

parser, 81

PayPal, 500

PercentMobile, 487

Perfecto Mobile, 474

pierwotny nagłówek, 482

ping time, 48

platforma DeviceAtlas, 164

plik

- Application.php, 164
- archive.php, 294
- box.tpl.php, 375
- category.php, 294
- comment.tpl.php, 376
- comments.php, 303
- cookie, 175
- da.php, 165
- default.php, 420
- desktop.css, 170
- detection.php, 161
- device_type, 177
- footer.php, 289
- functions.php, 296, 305
- gallery.html, 202
- header.php, 289, 444
- headers.php, 157
- huge.css, 220
- index.php, 169, 417
- iwebkit.html, 239
- medium.css, 223
- modul_mobilny.info, 380
- modul_mobilny.module, 381

narrow.css, 220, 222

node.tpl.php, 353, 374, 390

node-produkt.tpl.php, 390

page.php, 298

page.tpl.php, 353

redirect.php, 175

registration.html, 181

responsive/medium.css, 224

responsive/narrow.css, 222

responsive/narrow-medium-wide.css, 226

responsive2/Index.php, 233

screenshot.png, 270, 353

sencha.html, 248

single.php, 298, 448

skorka_mobilna.info, 352

style.css, 181

style.html, 182

switcher.php, 173, 433

szablonu, 293

templateDetails.xml, 416, 430, 433

theme.php, 168, 169, 173

wtyczka-mobilna.php, 308, 443

wurfl-config.xml, 163

pliki

- .info, 352

- dziennika, 484

- szablonów, 294

pobieranie treści, 107

podłączanie emulatora do debugera, 470

podział

- na kategorie, 122

- na strony, 192, 314

- strony, 141

- urządzeń na grupy, 228

pojemność pamięci przeglądarki, 140

pole textarea, 204

pole Zapamiętaj mnie, 186

polecenie

- ROBOTEXCLUDE, 486

- ROBOTINCLUDE, 486

połączenie między przeglądarką a debugerem, 469

port HTTP, 51

port serwera WWW, 51

porty TCP/IP, 51

programy operatorów sieci, 78

projekt mobilny, 216, 228

projektowanie, 207

projektowanie formularzy, 180

projektowanie interfejsów mobilnych, 214

projektowanie pod kątem grup urządzeń, 228

- projekty oparte na kliencie, 216
- projekty oparte na serwerze, 228
- protokół
 - HTTP, 108
 - RPC, 286
 - WAP, 28, 155, 161
- przechodzenie między stronami, 298
- przechowywanie danych, 74
- przeglądarka
 - Chromium, 457
 - Firefox, 452
 - Google Chrome, 42, 457
 - Internet Explorer, 42
 - Mobile Safari, 71
 - Mozilla Firefox, 42
 - Opera, 44, 458, 468
 - Opera Mini, 44
 - Opera Mobile, 44
 - Safari, 457
 - Safari Apple, 42
- przeglądarka mobilna
 - Android, 87
 - BlackBerry, 91
 - Dolfin, 90
 - MicroB, 94
 - Mobile Internet Explorer, 91
 - Mobile Safari, 82
 - Mozilla Fennec, 240
 - Opera Mini, 92
 - Opera Mobile, 92
 - Ovi, 94
 - PalmOS, 91
 - wrsje Nokii, 89
- przeglądarki, 44
 - generujące ruch w witrynie, 485
 - mobilne, 81
 - używane do wyświetlania witryny, 485
- przekierowania, 341
- przekierowywanie niespójnych żądań, 130
- przełączanie między motywami, 175
- przełączanie motywów, 167, 310
- przełączanie skórek, 341
- przełączanie wersji witryn, 137
- przełącznik, 257
- przełącznik agenta, 275
- przenoszenie zarządzanych treści, 98
- przepisywanie zawartości, 436
- przepustowość, 48
- przycisk Back, 244
- Pulsar, 112

- punkt podziału, 400
- punkty dostępowe WiFi, 60
- punkty wejścia, 126

R

- rdzeń, 99
- reguły wykrywania urządzeń, 332, 333
- rejestracja, 179
- rejestracja funkcji, 309
- rejestracja skryptu, 465
- reklama, 496
- reklama tekstowa, 500
- rodzaje reklam, 497
- rodzaje odnośników, 133
- rodzaje zawartości
 - Page, strona, 322
 - Story, artykuł, 322
- rola, 343
- rozdzielczość wyświetlaczy, 65
- rozszerzenie
 - Auto Template Switcher, 401
 - Mobilebot, 402, 406
- rozwijanie, 198
- rozwijanie elementu wewnątrzwierszowo, 197
- rysik, 42
- rysunek, 316
 - skalowanie, 318
 - tinySrc, 318
 - zmienianie wielkości, 318
- rzeczywistość rozszerzona, 60

S

- Safari
 - inspektor, 457
 - menu Programowanie, 458
- SDK, Software Development Kit, 112
- segment
 - \$comment_count, 360
 - \$content, 360
 - \$date, 360
 - \$links, 360
 - \$name, 360
 - \$node_url, 360
 - \$picture, 360
 - \$terms, 360
 - \$title, 360
 - \$type, 360

- segment, node, 322
- sekcja, 394
- Sencha Touch, 246
- serwer Apache, 108
- serwer WWW, 108
- serwis
 - DeviceAnywhere, 117
 - Forum Nokia, 117, 474
 - foursquare, 105
 - mobiForge, 78
 - NFL.com, 211
 - testowy mobiReady, 476
 - TinySrc, 203
- sieci do przesyłu danych, 47
- sieci reklamowe
 - AdMob, 497
 - Google AdSense, 499
- sieć
 - EDGE, 47, 49
 - ESPN, 208
 - GPRS, 49
 - HSDPA, 49
 - iMode, 493
 - mobilna, 47
 - PSDN, 27
 - trzeciej generacji (3G), 29
 - UMTS, 47
 - WiMAX, 68
 - WWW, 32
- silnik Gecko, 94
- silnik Trident, 92
- silnik WebKit, 81, 457
- skalowanie rysunków, 203, 226
- sklep
 - App Store, 53
 - Ovi Nokii, 54
- skórka, 352
 - Adaptivetheme Mobile, 350
 - iDrupal, 349
 - mobile_garland, 346
 - mobilna, 345, 352
- skórki
 - Drupala, 325
 - Nokii, 346
- skrypty, 73
- SMS, Short Message Service, 57
- specyfikacja CSS3, 217
- specyfikacja stylów, 72
- spójność marki, 35
- spójność tematyczna, 34
- sprawdzanie nagłówek, 154
- sprzedaż usług, 500
- SSI, Server-Side Includes, 96
- standard
 - BONDI, 56
 - GSM, 26
 - SVG, 74, 144
 - WAP, 44
- stopniowe wzbogacanie, progressive enhancement, 220
- stosowanie AJAX-a, 144, 150
- strona
 - do edycji bloków, 369
 - do konfigurowania skórki, 353
 - po dodaniu bloków, 370
 - po zastosowaniu stylów, 372
 - z logo, 294
 - z menu dodatkowym, 368
 - z motywem mobilnym, 312
 - z podstawowymi stylami, 292
 - z prostym stylem, 222
 - z ustawieniami grup urządzeń, 329
 - ze stylami, 360
- struktura systemu CMS, 100
- styl formularza, 377
- style, 71
- style CSS, 147
- style mobilne, 205
- SVG, Scalable Vector Graphics, 144
- SVG, Scalar Vector Graphics, 74
- Symbian, 68
- symulator
 - BlackBerry, 114
 - Opera, 114
 - Opera Mini, 115, 471
 - systemu iOS, 459
- system CMS, 98
 - administrator, 99
 - baza danych, 98
 - motywy, 99
 - rdzeń, 99
 - wtyczki, 99
- system Drupal, *Patrz* Drupal
- system GSM, 47
- system iOS4.2, 82
- system operacyjny
 - Android, 69
 - BlackBerry OS, 69, 464
 - iOS, 69
 - MeeGo, 69, 90, 94

system operacyjny
 Symbian, 29, 68
 Web OS, 69, 91, 466
 Windows Mobile 2008, 92
 Windows Mobile 7, 90
 Windows Phone, 69

system Joomla!, *Patrz Joomla!*
 system WordPress, *Patrz WordPress*

systemy CMS, 251
 Drupal, 251, 321
 Joomla!, 251, 393
 WordPress, 251

szablon TapTheme, 413

szablon WAFLI, 398

szablony internetowe Nokii, 240

szerokość ekranu, 221, 231

szkielet motywu, 293

Ś

ścieżka powrotu, Show pathway, 134, 410

środowisko programistyczne Eclipse, 78, 461

Pulsar, 112

środowisko IDE, 110

T

tabele, 240

tablica \$_SERVER, 158

TapTheme, 411–413

technika CAPTCHA, 185

technika SSI, 97

technologia

BREW, 53

Flash, 74

NFC, 67

transkodowania, 49–50, 481

WebSocket, 75

technologie sieciowe, 67

testowanie, 115, 451

interfejsów mobilnych, 458

rzeczywistych urządzeń, 480

urządzeń przenośnych, 473

witryn, 480

transkoder, 49, 482

transkodowanie, 49–50, 481

treść, 240

triangulacja, 59

tworzenie

dotatku dla systemu Joomla!, 432

dostosowujących się projektów, 217

modułów Drupala, 380

stylów dekoracji, 195

wersji mobilnej, 104

witryny, 167

U

uchwyt, hook, 307

hook_init(), 381

hook_nodeapi(), 383

układ strony, 288

umieszczanie modułów w szablonie, 409

unikanie transkodowania, 482

urządzenia przenośne, 64

aparaty, 67

film, 67

jednobryłowe dotykowe, 38

jednobryłowe tradycyjne, 38

lokalizacja i ułożenie, 67

natywne funkcje, 75

rozsuwane, 39

szerokość wyświetlaczy, 43

szybkość procesora, 45

technologia NFC, 67

wyświetlacze, 64

z klapką, 40

zużycie energii, 45

urządzenia wejścia, 66

urządzenie BlackJack II, 165

User-Agent Switcher, 452

usługa

Express Checkout, 500

i-mode, 27

mobileOK, 496

OSMOBI, 395

PercentMobile, 487

raport, 488

raport szczegółowy, 489

ready.mobi, 496

tinySrc, 318

ustawienia

domen, 328

grupy mobile, 330

motywu mobilnego, 267

nowej grupy, 330

przełącznika, 261

Both browser detection, 261

Browser detection, 261

Disabled, 261

domain mapping, 261

Domain mapping, 261

- rozszerzenia WAFL, 397
- skórki mobilnej, 337
- użytkowników, 365
- w zakładce Device Groups, 347
- wtyczki WPTouch, 278
- wyświetlania, 266
- usunięcie filmów wideo, 314
- użytkownicy mobilni, 34, 104
- użytkownicy stacjonarni, 104

V

- VirtualBox, 466
- Visual Studio, 78

W

- W3C, World Wide Web Consortium, 72, 115, 477
- WAC, Wholesale Applications Community, 55, 76
- WAFL, Website Adaptation and Formatting Layer, 395
- waga reguły, 332
- walidator, 115
- walidator dotMobi, 117
- walidator W3C, 116, 477
- WAP, 159
- WebKit, 70
- wersja mobilna, 138
- wersja mobilna witryny, 96
- wersja stacjonarna, 138
- wget, 454
- WHATWG, 71
- widżet
 - Mobile Ads, 264
 - Mobile Barcode, 263
 - Mobile Switcher Link, 263
 - Mpexo, 264
- widżety, 263
- widżety dla urządzeń mobilnych, 56
- widżety w motywie domyślnym, 265
- wiersz poleceń, 454
- WiFi, 68
- WiMAX, 48
- Windows Mobile, 470
- Windows Phone, 69
- witryna, 138
 - GSM Arena, 78
 - mobilna, 96, 128, 213
 - Quirksmode, 150
 - stacjonarna, 128
 - statyczna, 96
- witryny z funkcjami handlu elektronicznego, 500
- włączanie modułów, 388
- WML, 27, 155, 405
- WordPress, 253
 - dodawanie komentarzy, 301
 - galeria, 255
 - komentarze, 255
 - motywy, 254, 256
 - odnośniki, 256
 - pliki szablonów, 298
 - przesyłanie plików, 255
 - strony, 254
 - struktura bloga, 255
 - struktura witryny, 254
 - szablony motywów, 293
 - widżety, 256
 - wpisy, 254
 - wtyczki, 254, 257
- WordPress Mobile (base), 269
- WordPress Mobile Edition, 281
 - instalowanie, 281
 - konfiguracja, 283
 - motyw, 281
- wpis z kategoriami i tagami, 300
- wpisy dziennika, 486
- WPTouch, 274
 - instalowanie, 275
 - konfiguracja, 277
 - listy, 276
 - menu My Account, 279
 - mobilne reklamy, 281
 - motyw, 275
 - ustawienia stylów i kolorów, 279
- wskazywanie skórki, 345
- współużytkowanie danych, 106
- wtyczka
 - dotMobi WordPress Mobile Pack, 257
 - MobilePress, 283
 - WordPress Mobile Edition, 281
 - WordPress Mobile Pack, 498
 - WPTouch, 274
- wtyczki, 99, 257
- WURFL, 163
- wybieranie
 - motywu, 173, 310
 - wersji, 154
 - widżetów, 266

- wykrywanie grup, 230
- wykrywanie przeglądark, 153, 161
 - algorytm, 161
 - na podstawie bazy danych, 163
 - oparte na nagłówkach, 163
 - po stronie klienta, 165
- wyniki testów, 477
- wyniki w narzędziu mobileOK Checker, 479
- wyniki wyświetlane przez walidator, 479
- wyszukiwanie, 492
- wyszukiwanie wzbogacone, 200
- wyszukiwarka Bing, 493
- wyszukiwarka Google, 493
- wyszukiwarka mobilna, 492
- wyszukiwarka Yahoo!, 493
- wyświetlanie stron, 298
- wyświetlanie wpisów, 298
- wzbogacanie motywów, 269
- wzorzec projektowy, 212

X

- XHTML, 31, 46
- XHTML for Mobile, 77

Z

- zabezpieczanie witryny, 481
- zapis motywów, 260
- zapytania Media Query, 217
- zarabianie na witrynie, 496
- zarządzanie blokami, 331
- zarządzanie widżetami, 264
- zdalnie urządzenie Nokii, 475
- zintegrowane środowiska programowania, IDE, 111
- zmiany stylów, 216

- zmienna
 - \$base_path, 355
 - \$breadcrumb, 355
 - \$browser_type, 172
 - \$closure, 355
 - \$content, 354
 - \$footer, 356
 - \$footer_message, 355
 - \$head, 355
 - \$head_title, 354
 - \$header, 356
 - \$help, 355
 - \$left, 356
 - \$logo, 355
 - \$messages, 355
 - \$mission, 355, 356
 - \$node, 387
 - \$primary_links, 355
 - \$right, 356
 - \$scripts, 355
 - \$search_box, 355
 - \$secondary_links, 355
 - \$site_name, 355
 - \$site_slogan, 355
 - \$styles, 355, 358
 - \$tabs, 355
 - \$this, 421
 - \$title, 355
- znaczniki, 70
- znaczniki dla profilu mobilnego, 71
- znaczniki galerii, 201
- zwijanie, 197

PROGRAM PARTNERSKI

GRUPY WYDAWNICZEJ HELION



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

Zmień swoją stronę WWW
w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

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

GRUPA WYDAWNICZA

 **Helion SA**

Programowanie

mobilnych stron internetowych z wykorzystaniem systemów CMS

Świat oszalał na punkcie smartfonów, które wreszcie umożliwiły wygodne, szybkie przeglądanie Internetu i sprawiły, że korzystanie z zasobów sieci w telefonie stało się tak samo naturalną czynnością jak wykonywanie połączeń. To oczywiście zrodziło potrzebę tworzenia lżejszych, lecz tak samo funkcjonalnych oraz atrakcyjnych wizualnie odpowiedników różnych serwisów, e-sklepów i aplikacji, zapewniających komfortowe korzystanie z ich stron na wszelkich urządzeniach mobilnych.

Oto książka, w której znajdziesz szczegółowe omówienie technologii Internetu mobilnego – technik i narzędzi, które można wykorzystać do udostępnienia na urządzeniach przenośnych rozmaitych materiałów sieciowych. Przedstawiono tu standardowe wzorce rozwijania interfejsów użytkownika, a także ułatwiające pracę szablony i platformy. Autor koncentruje się na trzech najważniejszych CMS-ach (WordPress, Joomla! oraz Drupal) i wyjaśnia, jak projektować, budować oraz publikować za ich pomocą użyteczne witryny, które zachwycą użytkowników mobilnego Internetu!

W książce znajdziesz między innymi:

- ▶ **techniczne omówienie środowiska mobilnego oraz wprowadzenie do urządzeń i sieci mobilnych, a także związanych z nimi problemów;**
- ▶ **opis ewolucji Internetu mobilnego i optymalnych sposobów dostosowania się do niej;**
- ▶ **omówienie wyborów i decyzji, które trzeba podjąć przed udostępnieniem witryny opartej na systemie CMS w Internecie mobilnym;**
- ▶ **opis procesów związanych z testowaniem, instalowaniem i integrowaniem witryn mobilnych w dowolnym systemie;**
- ▶ **omówienie analizy, reklamy i innych zagadnień z obszaru mobilnego.**

Wydź naprzeciw możliwościom, jakie oferują Ci popularne platformy CMS, i twórz niezwykłe witryny z myślą o mobilnym Internecie.

James Pearce jest specjalistą od technologii, autorem książek, programistą i przedsiębiorcą pracującym w branży Internetu mobilnego od ponad dziesięciu lat. Jest także autorem popularnej wtyczki WordPress Mobile Pack.

helion.pl
księgarnia
internetowa

Nr katalogowy: **7848**



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

<http://helion.pl>

sięgnij po **WIEGE!**



800 korzyści

ISBN 978-83-246-3610-5



Cena 89,00 zł

informatyka w najlepszym wydaniu

9 788324 636105