



Wtyczki do

# WordPressa

Programowanie dla profesjonalistów

Tytuł oryginału: Professional WordPress Plugin Development

Tłumaczenie: Robert Górczyński

ISBN: 978-83-246-3564-1

© 2011 by Wiley Publishing, Inc., Indianapolis, Indiana

All Rights Reserved. Authorized translation from the English language edition published by John Wiley & Sons Limited. Responsibility for the accuracy of the translation rests solely with Helion S. A. and is not the responsibility of John Wiley & Sons Limited. No part of this book may be reproduced in any form without the written permission of the original copyright holder, John Wiley & Sons Limited.

Wiley, the Wiley logo, Wrox, the Wrox logo, Wrox 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. 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.

Translation copyright © 2012 by Wydawnictwo Helion.

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

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

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

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](mailto:helion@helion.pl)

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

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

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

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

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

# Spis treści

<b>O autorach .....</b>	<b>15</b>
<b>Wstęp .....</b>	<b>17</b>
<b>Wprowadzenie .....</b>	<b>19</b>
<b>Rozdział 1. Wprowadzenie do wtyczek .....</b>	<b>23</b>
Co to jest wtyczka? .....	23
W jaki sposób wtyczki współdziałają z platformą WordPress? .....	24
Kiedy wtyczki są wczytywane? .....	25
Dostępne wtyczki .....	25
Oficjalny katalog wtyczek .....	26
Przykłady popularnych wtyczek .....	26
Popularne tagi wtyczek .....	27
Zalety wtyczek .....	27
Brak konieczności modyfikacji jądra platformy .....	27
Nie trzeba wyważać otwartych drzwi .....	28
Oddzielenie wtyczek i motywów .....	28
Łatwe uaktualnienia .....	29
Łatwiejsze dzielenie się wtyczkami i ich ponowne używanie .....	29
Wtyczki są oddzielone od siebie .....	29
Społeczność tworząca wtyczki .....	30
Instalacja wtyczek i zarządzanie nimi .....	30
Instalacja wtyczki .....	30
Zarządzanie wtyczkami .....	31
Edycja wtyczek .....	31
Katalog wtyczek .....	32
Typy wtyczek .....	32
Testowanie funkcji wtyczek .....	33
Podsumowanie .....	34
<b>Rozdział 2. Podstawy wtyczek .....</b>	<b>35</b>
Utworzenie pliku wtyczki .....	35
Nadanie nazwy wtyczce .....	35
Używanie katalogu .....	36
Stosowanie rozsądnych praktyk .....	36
Stosowanie prefiksu w każdej sytuacji .....	36
Organizacja pliku .....	37
Struktura katalogów .....	37

Wymagania dotyczące nagłówka .....	38
Utworzenie nagłówka .....	38
Licencja wtyczki .....	39
Określanie ścieżek dostępu .....	39
Ścieżki dostępu wtyczki .....	40
Lokalne ścieżki dostępu .....	40
Adresy URL .....	41
Aktywacja i dezaktywacja funkcji .....	42
Funkcja aktywacji wtyczki .....	42
Utworzenie ustawień domyślnych podczas aktywacji .....	43
Funkcja dezaktywacji wtyczki .....	43
Dezaktywacja to nie dezinstalacja wtyczki .....	44
Metody dezinstalacji .....	44
Dlaczego dezinstalacja wtyczki jest konieczna? .....	44
Plik uninstall.php .....	44
Zaczep uninstall .....	45
Standardy tworzenia kodu .....	46
Twórz dokumentację kodu .....	46
Nazwy zmiennych, funkcji i plików .....	47
Apostrof i cudzysłów .....	47
Wcięcia .....	48
Styl stosowania nawiasów .....	48
Używanie spacji .....	49
Skrócone znaczniki PHP .....	49
Polecenia SQL .....	49
Lista rzeczy do sprawdzenia podczas prac nad wtyczkami .....	49
Podsumowanie .....	50

### **Rozdział 3. Zaczepy ..... 51**

Akcje .....	52
Czym jest akcja? .....	53
Funkcje zaczepu akcji .....	54
Najczęściej używane zaczepy akcji .....	58
Filtry .....	61
Funkcje zaczepu filtru .....	63
Funkcje szybko zwracające wartość .....	67
Najczęściej używane zaczepy filtru .....	68
Używanie zaczepów z poziomu klasy .....	71
Tworzenie własnych zaczepów .....	72
Zalety utworzenia własnego zaczepu .....	73
Przykład utworzenia własnego zaczepu akcji .....	73
Przykład własnego zaczepu filtru .....	73
W jaki sposób wyszukiwać zaczepy? .....	75
Wyszukiwanie zaczepów w kodzie tworzącym jądro WordPress .....	75
Zaczepy zmienne .....	75
Listy zaczepów .....	76
Podsumowanie .....	76

<b>Rozdział 4. Integracja z platformą WordPress .....</b>	<b>77</b>
Dodawanie menu i podmenu .....	77
Utworzenie menu najwyższego poziomu .....	77
Dodawanie podmenu .....	78
Dodawanie elementu menu do już istniejącego menu .....	80
Tworzenie widgetów .....	82
Utworzenie widgetu .....	82
Widget zaawansowany .....	87
Tworzenie widgetów kokpitu .....	92
Utworzenie widgetu kokpitu wraz z opcjami .....	93
Pola użytkowników .....	96
Dodawanie własnego pola użytkownika .....	97
Zapis danych pola użytkownika .....	97
Zaawansowane pole użytkownika .....	101
Zachowanie spójności .....	106
Korzystanie z interfejsu użytkownika platformy WordPress .....	106
Podsumowanie .....	113
<b>Rozdział 5. Internacjonalizacja .....</b>	<b>115</b>
Internacjonalizacja i tłumaczenie na inne języki .....	115
Dlaczego warto przeprowadzać internacjonalizację? .....	116
Zrozumienie zagadnienia internacjonalizacji w profesjonalnej pracy .....	116
Przygotowanie wtyczki do tłumaczenia na inne języki .....	117
Wyświetlanie i zwracanie ciągów tekstowych .....	118
Używanie miejsc zarezerwowanych .....	125
Internacjonalizacja kodu JavaScript .....	127
Tworzenie plików tłumaczenia .....	130
Pliki MO i PO .....	130
Narzędzia służące do tłumaczenia .....	130
W jaki sposób utworzyć plik POT? .....	131
Gdzie przechowywać pliki tłumaczeń? .....	131
Podsumowanie .....	132
<b>Rozdział 6. Bezpieczeństwo wtyczki .....</b>	<b>133</b>
Zabezpieczenie wtyczki .....	133
Czym jest zapewnienie bezpieczeństwa wtyczce? .....	134
Czym nie jest zapewnienie bezpieczeństwa wtyczce? .....	134
Uprawnienia użytkownika .....	134
W jaki sposób używać funkcji <code>current_user_can()</code> ? .....	134
Nie sprawdzaj zbyt wcześnie .....	135
Unikalne identyfikatory .....	136
Uprawnienia kontra zamiary .....	136
Czym jest unikalny identyfikator? .....	137
Jak tworzyć i weryfikować unikalne identyfikatory? .....	137
Unikalne identyfikatory w skryptach Ajax .....	142
Weryfikacja i oczyszczenie danych .....	143
Potrzeba weryfikacji i oczyszczania danych .....	143
Dobra praktyka: identyfikacja potencjalnie niebezpiecznych danych .....	144
Weryfikacja czy oczyszczanie danych wejściowych? .....	146
Przykłady weryfikacji i oczyszczania danych .....	147

Formatowanie poleceń SQL .....	163
Obiekt \$wpdb .....	163
Dlaczego metody obiektu wpdb są lepsze? .....	163
Metody typu „wszystko w jednym” .....	164
Najczęściej stosowane metody .....	166
Ochrona zapytań przed atakami typu SQL Injection .....	170
Różne metody i właściwości obiektu wpdb .....	172
Dobre nawyki bezpieczeństwa .....	172
Podsumowanie .....	174
<b>Rozdział 7. Ustawienia wtyczki .....</b>	<b>175</b>
API Options .....	175
Zapisywanie opcji .....	175
Zapisywanie tablicy opcji .....	176
Pobieranie opcji .....	177
Wczytywanie tablicy opcji .....	178
Usuwanie opcji .....	178
Parametr autoload .....	179
API Settings .....	180
Zalety API Settings .....	181
Funkcje API Settings .....	181
Zebranie całości: pełna strona zarządzania wtyczką .....	185
Usprawnienie reakcji funkcji i weryfikacja błędów .....	187
Dodawanie pól na istniejącej stronie .....	188
API Transients .....	192
Zapisywanie opcji, która ma utracić ważność .....	192
Pobieranie opcji, która ma utracić ważność .....	192
Usunięcie opcji, która utraciła ważność .....	192
Praktyczny przykład użycia krótkotrwałych danych .....	193
Szczegółowe informacje techniczne .....	193
Idea krótkotrwałych danych .....	193
Zapisywanie ustawień poszczególnych użytkowników .....	194
Tworzenie wtyczki .....	194
Metadane użytkownika .....	194
Uaktualnianie metadanych użytkownika .....	195
Pobieranie metadanych użytkownika .....	196
Usunięcie metadanych użytkownika .....	196
Pobieranie identyfikatora użytkownika .....	197
Dodawanie pól na stronie profilu .....	197
Wtyczka BOJ Admin Lang .....	199
Ustawienia dla poszczególnych użytkowników — najlepsze praktyki .....	200
Przechowywanie danych we własnych tabelach .....	201
Typy danych .....	201
Standardowe tabele WordPress .....	202
Tworzenie własnej tabeli .....	202
Uaktualnienie struktury własnej tabeli .....	203
Uzyskanie dostępu do własnej tabeli .....	206
Podsumowanie .....	207

<b>Rozdział 8. Użytkownicy .....</b>	<b>209</b>
Praca z użytkownikami .....	210
Funkcje użytkownika .....	210
Tworzenie, uaktualnianie i usuwanie użytkowników .....	214
Dane użytkownika .....	218
Metadane użytkownika .....	223
Role i możliwości .....	229
Czym są role i możliwości? .....	230
Role domyślne .....	230
Własne role .....	231
Ograniczanie dostępu .....	231
Sprawdzanie uprawnień użytkownika .....	232
Czy użytkownik jest administratorem? .....	236
Nadanie własnych uprawnień .....	237
Dostosowanie ról do własnych potrzeb .....	238
Tworzenie roli .....	238
Usunięcie roli .....	239
Dodanie możliwości do roli .....	241
Usuwanie możliwości z roli .....	241
Wtyczka obsługująca własne role i możliwości .....	242
Podsumowanie .....	245
<b>Rozdział 9. API HTTP .....</b>	<b>247</b>
Szybki kurs wykonywania żądań HTTP .....	247
Czym jest żądanie HTTP? .....	247
Jak wykonywać żądania HTTP w PHP? .....	250
Funkcje obsługi HTTP oferowane przez WordPress .....	251
Funkcje rodziny wp_remote_* .....	252
Konfiguracja zaawansowana i wskazówki .....	257
Ćwiczenie praktyczne: odczyt formatu JSON z zewnętrznego API .....	263
Pobieranie i odczytywanie danych JSON .....	263
Funkcjonująca wtyczka .....	264
Ćwiczenie praktyczne: wysyłanie danych do zdalnego API .....	267
Formatowanie parametrów dla żądań POST .....	267
Gotowa wtyczka .....	268
Ćwiczenie praktyczne: odczyt dowolnej treści .....	269
Utworzenie własnego repozytorium wtyczki .....	270
Jak działa proces uaktualnienia wtyczki na platformie WordPress? .....	270
Wykonywanie żądań do alternatywnego API z poziomu wtyczki .....	272
Utworzenie alternatywnego API .....	274
Kilka ostrzeżeń dotyczących własnych API .....	276
Przypadek specjalny: pobieranie zdalnych wiadomości RSS .....	276
Podsumowanie .....	277
<b>Rozdział 10. API Shortcode .....</b>	<b>279</b>
Tworzenie skrótów .....	279
Czym jest skrót? .....	279
Rejestracja własnego skrótów .....	280
Wskazówki dotyczące skrótów .....	284
Pomyśl o prostocie .....	284
Pamiętaj o dynamiczności .....	287

Wewnętrzny sposób działania .....	288
Kod BBCode we wtyczce obsługującej komentarze .....	290
Ograniczenia skrótów podczas obsługi struktur zagnieżdżonych .....	292
Integracja z usługą Google Mapy .....	293
Uzyskanie dostępu do API Google Geocoding .....	293
Przechowywanie wyników .....	295
Uzyskanie dostępu do API Google Maps .....	296
Więcej pomysłów dotyczących skrótów .....	300
Wyświetlanie treści jedynie dla zalogowanych użytkowników .....	301
Wyświetlenie treści ograniczonej czasowo .....	302
Zaciemnienie adresu e-mail .....	302
Podsumowanie .....	303

## **Rozdział 11. Rozbudowa wpisów bloga: metadane, własne typy wpisów bloga i taksonomie .....305**

Tworzenie własnych typów wpisów bloga .....	306
Możliwe typy wpisów bloga .....	306
Rejestracja typu wpisu bloga .....	306
Ustawianie etykiet we własnym typie wpisu bloga .....	311
Wykorzystanie własnych możliwości .....	313
Dołączanie istniejących taksonomii .....	314
Używanie własnych typów wpisów bloga .....	315
Utworzenie pętli własnego typu wpisu bloga .....	315
Pobieranie treści własnego typu wpisu bloga .....	317
Sprawdzenie istnienia typu wpisu bloga .....	318
Metadane wpisu bloga .....	319
Dodawanie metadanych wpisu bloga .....	320
Pobieranie metadanych .....	321
Uaktualnienie metadanych wpisu bloga .....	321
Usuwanie metadanych .....	322
Tworzenie własnych taksonomii .....	323
Zrozumienie taksonomii .....	323
Rejestracja własnej taksonomii .....	324
Przypisanie taksonomii do typu wpisu bloga .....	329
Używanie własnych taksonomii .....	329
Pobieranie taksonomii .....	329
Używanie taksonomii wraz z wpisami bloga .....	330
Tagi warunkowe taksonomii .....	332
Wtyczka typu wpisu bloga oraz taksonomii .....	334
Podsumowanie .....	336

## **Rozdział 12. Technologie JavaScript i Ajax na platformie WordPress .....337**

Krótkie wprowadzenie do jQuery .....	337
Zalety wynikające z używania jQuery .....	337
Krótki kurs jQuery .....	338
Technologia Ajax .....	341
Czym jest Ajax? .....	341
Najlepsze praktyki dotyczące technologii Ajax .....	344
Dodawanie kodu JavaScript do WordPress .....	345
Prawidłowy sposób dołączania skryptów .....	345
Gdzie umieszczać skrypty? .....	351



Dodawanie skryptów jedynie wtedy, gdy są potrzebne .....	353
Skrypty dynamiczne na platformie WordPress .....	357
Technologia Ajax na platformie WordPress .....	360
Technologia Ajax na platformie WordPress: reguły .....	360
Kompletny przykład: natychmiastowe odnośniki „Czytaj dalej” .....	362
Kolejny przykład: usunięcie komentarza ze strony .....	369
Usuwanie błędów podczas używania technologii Ajax .....	373
Podsumowanie .....	374
<b>Rozdział 13. Cron .....</b>	<b>375</b>
Czym jest cron? .....	375
W jaki sposób działa demon cron? .....	375
Tworzenie harmonogramu zadań cron .....	376
Utworzenie powtarzającego się zadania harmonogramu .....	376
Utworzenie jednorazowego zadania harmonogramu .....	378
Usunięcie zadania z harmonogramu .....	380
Zdefiniowanie własnych odstępów czasu .....	381
Wyświetlenie zadań harmonogramu cron .....	381
Prawdziwy cron .....	385
Przykłady praktyczne .....	385
Usuwanie co tydzień wcześniejszych wersji wpisu bloga .....	385
Wtyczka automatycznie wysyłająca wiadomość e-mail .....	390
Wtyczka usuwająca komentarze .....	393
Podsumowanie .....	399
<b>Rozdział 14. API Rewrite .....</b>	<b>401</b>
Dlaczego czasem trzeba zmieniać adresy URL? .....	401
Zasady dotyczące odnośników bezpośrednich .....	402
Moduł mod_rewrite serwera Apache .....	402
Zmiany adresów URL na platformie WordPress .....	403
W jaki sposób WordPress obsługuje zapytania? .....	404
Ogólny opis procesu wykonania zapytania .....	404
Obiekt rewrite .....	405
Obiekt query .....	405
Co można zrobić przy użyciu wtyczek? .....	406
Przykłady praktyczne .....	406
Zmiana adresu URL w celu utworzenia listy sklepów .....	407
Tworzenie nowej struktury odnośników bezpośrednich oraz integracja ze stronami, które nie powstały w WordPress .....	412
Wyświetlanie produktów sklepu .....	414
Dodawanie punktu końcowego i zmiana formatu danych wyjściowych .....	415
Dodanie własnego kanału wiadomości informującego o ostatnio dodanych obrazach .....	419
Podsumowanie .....	421
<b>Rozdział 15. Sieć Multisite .....</b>	<b>423</b>
Różnice .....	424
Standardowa konfiguracja WordPress kontra sieć Multisite .....	424
Zrozumienie terminologii sieci Multisite .....	424
Zalety sieci Multisite .....	425
Włączenie sieci Multisite na platformie WordPress .....	425

Funkcje sieci Multisite .....	427
Potęga identyfikatora bloga .....	427
Najczęściej używane funkcje .....	427
Przełączenie i przywracanie witryn internetowych .....	429
Przykłady skrótów uzyskania dostępu do treści sieci .....	432
Przykład widgetu z treścią sieciową .....	436
Utworzenie nowej witryny .....	443
Opcje witryny sieci Multisite .....	448
Użytkownicy w sieci .....	448
Rola Superadministratora w sieci Multisite .....	452
Sprawdzenie właściciela witryny .....	453
Dane statystyczne dotyczące sieci .....	454
Schemat bazy danych sieci Multisite .....	455
Tabele stosowane w sieci Multisite .....	455
Tabele przeznaczone dla konkretnych witryn .....	455
Podsumowanie .....	456

## **Rozdział 16. Usuwanie błędów i optymalizacja .....457**

Zapewnienie (lub nie) obsługi starszych wersji .....	457
Aktualizacja oprogramowania zgodnie z cyklem rozwojowym	
WordPress .....	458
Funkcje uznane za przestarzałe .....	459
Obsługa zbędnych instalacji .....	460
Usuwanie błędów .....	460
Włączenie trybu usuwania błędów .....	461
Wyświetlanie komunikatów związanych z usuwaniem błędów .....	461
Poprawianie błędów wskazywanych przez komunikaty .....	462
Rejestrowanie błędów .....	466
Włączenie rejestrowania błędów .....	466
Położenie pliku dziennika błędów .....	467
Plik dziennika błędów .....	467
Buforowanie .....	467
Zapisywanie, wczytywanie i usuwanie buforowanych danych .....	468
Buforowanie danych we wtyczce .....	469
Podsumowanie .....	471

## **Rozdział 17. Działania marketingowe .....473**

Wybór licencji dla wtyczki .....	474
Różne opcje .....	474
Dlaczego licencja ma znaczenie? .....	475
Zarabianie pieniędzy pomimo stosowania licencji GPL .....	476
Udostępnienie wtyczki na witrynie WordPress.org .....	477
Utworzenie konta .....	478
Zgłoszenie wtyczki do oficjalnego repozytorium .....	479
Konfiguracja SVN .....	479
Utworzenie pliku readme.txt .....	480
Rozślawienie wtyczki .....	483
Nadawanie nazwy wtyczce .....	483
Zbudowanie witryny internetowej .....	485
Utworzenie strony dla wtyczki .....	487
Ogłoszenie wydania wtyczki .....	487

---

Pomoc techniczna dla użytkowników wtyczki .....	488
Zbieranie informacji od użytkowników .....	488
Wyjście z piwnicy .....	490
Inne metody promocji .....	490
Podsumowanie .....	491
<b>Rozdział 18. Narzędzia programisty .....</b>	<b>493</b>
Jądro platformy jako punkt odniesienia .....	493
Dokumentacja osadzona na platformie .....	493
Wyszukiwanie funkcji .....	495
Najważniejsze pliki tworzące jądro platformy .....	495
Codex .....	497
Przeszukiwanie witryny Codex .....	497
Opis funkcji .....	498
Narzędzia oferowane przez inne witryny internetowe .....	498
PHPXref .....	498
Baza danych zaczepów platformy WordPress .....	500
Zasoby oferowane przez społeczność .....	500
Fora pomocy technicznej .....	500
Listy dyskusyjne .....	501
Czat WordPress .....	501
Informacje dotyczące prac rozwojowych nad WordPress .....	502
Zgłaszanie pomysłów dla WordPress .....	502
Obsługiwane przez społeczność witryny z nowościami .....	502
Wydarzenia lokalne .....	503
Narzędzia .....	503
Przeglądarka internetowa .....	504
Edytor tekstu .....	504
Obsługa plików za pomocą FTP, SFTP i SSH .....	505
phpMyAdmin .....	505
Podsumowanie .....	506
<b>Skorowidz .....</b>	<b>507</b>



# API Shortcode

## W TYM ROZDZIALE:

- Tworzenie własnego skrótu
- Rejestracja skomplikowanych i sparametryzowanych skrótów
- Zaawansowane wskazówki dotyczące skrótów
- Połączenie witryny z usługą Google Mapy

Skróty (ang. *shortcode*) to charakterystyczny dla WordPress kod pozwalający na wykonywanie różnych operacji przy minimalnym wysiłku, np. osadzanie treści lub tworzenie obiektów, które normalnie wymagałyby użycia dużej ilości skomplikowanego kodu.

W tym rozdziale dowiesz się, jak za pomocą jedynie niewielkiej liczby znaków umożliwić użytkownikom Twoich wtyczek rozbudowę wpisów bloga o zaawansowaną treść.

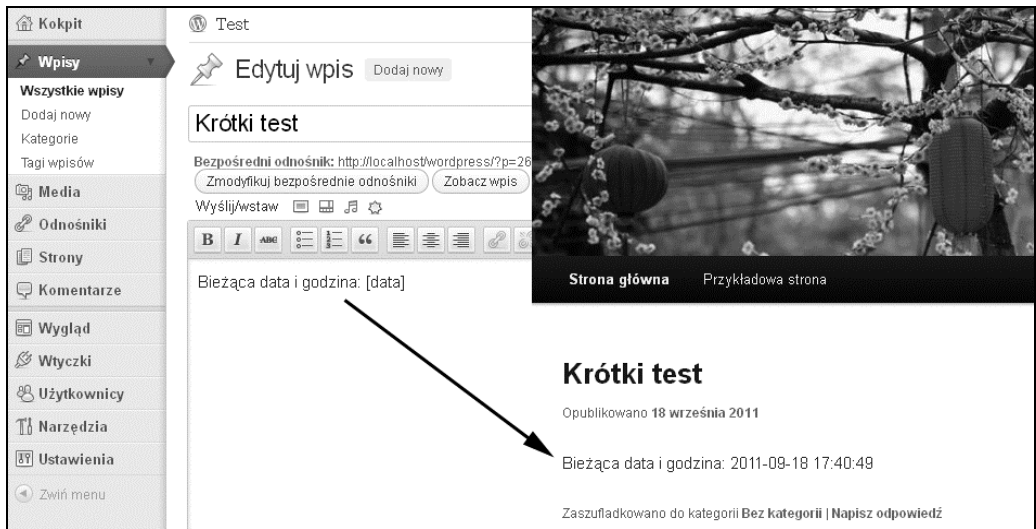
## Tworzenie skrótu

W tym podrozdziale przekonasz się, czym są skróty oraz jak można je tworzyć; zaczniemy od zamienników w postaci prostych ciągów tekstowych, a skończymy na funkcjach zaawansowanych wraz z parametrami.

## Czym jest skrót?

API Shortcode pozwala na tworzenie prostego kodu makro, czasami nazywanego BBCode z racji podobieństwa do popularnej składni w różnego rodzaju forach i serwisach komputerowych.

Ogólnie rzecz biorąc, skrót to prosta składnia znaczników w nawiasach kwadratowych, np. [znacznik], stosowanych we wpisach bloga. W trakcie generowania i wyświetlania takiego wpisu bloga skrót jest dynamicznie zastępowany bardziej skomplikowaną i zdefiniowaną przez użytkownika treścią. Na rysunku 10.1 pokazano przykład wtyczki skrótu, która znacznik [date] zastępuje bieżącą datą i godziną.



**Rysunek 10.1.** Wtyczka skrótu w działaniu

Platforma WordPress domyślnie rejestruje wiele skrótów gotowych do wykorzystania. Przykładowo podczas wysyłania do serwera wielu obrazów dołączonych do danego wpisu bloga można w nim umieścić skrót `[gallery]`, co spowoduje zastąpienie obrazów ładnie sformatowaną galerią.

Z technicznego punktu widzenia skrót może być dowolnym ciągiem tekstowym odpowiednim do użycia w charakterze klucza tablicy. Przykładowo wymienione poniżej ciągi tekstowe można zastosować jako skróty:

- `[foo]`,
- `[Foo]`,
- `[123]`,
- `[133t]`,
- `[Witaj Jestem Jan Kowalski]`.

W praktyce, w celu zachowania prostoty i uniknięcia potencjalnych konfliktów pomiędzy różnymi skrótami skróty będziesz rejestrował jako proste ciągi tekstowe zapisane małymi literami.



#### **UWAGA**

Nie wolno rejestrować własnych skrótów o nazwach `[wp_caption]`, `[caption]`, `[gallery]` i `[embed]`, ponieważ są już zarejestrowane przez WordPress.

## Rejestracja własnego skrótu

W tym punkcie dowiesz się, w jaki sposób rejestrować własne skróty. Ponadto poznasz praktyczne sposoby ich wykorzystania, od prostego zastępowania tagów aż po bardziej skomplikowane i sparametryzowane dane wyjściowe.

Przedstawione poniżej wtyczki skrótów używają prefiksu w postaci `boj_sxX`, gdzie X oznacza liczbę.

## [ksiazka]

Celem użycia skrótu jest przyspieszenie wprowadzania danych i zastąpienie często stosowanych zdań ich skrótami, łatwymi do zapamiętania i szybszymi do wpisania.

Jeżeli np. często wspominasz tytuł książki, którą promujesz w serwisie Amazon, wówczas zamiast za każdym razem pisać `<a href="http://www.amazon.com/dp/0470560541">książka</a>`, o wiele szybciej i prościej będzie napisać po prostu `[ksiazka]`, prawda?

W tym celu trzeba użyć funkcji `add_shortcode()` wymagającej podania dwóch parametrów:

- wzorca znacznika (bez otaczających go nawiasów kwadratowych);
- wywoływanej funkcji odpowiedzialnej za zastąpienie znacznika.



```
<?php
```

```
/*
```

```
Plugin Name: Przykład skrótu nr 1
```

```
Plugin URI: http://przyklad.pl/
```

```
Description: Zastępuje znacznik [ksiazka] długim odnośnikiem prowadzącym do serwisu Amazon.
```

```
Version: 1.0
```

```
Author: Ozh
```

```
Author URI: http://wrox.com/
```

```
*/
```

```
// Rejestracja nowego skrótu: [ksiazka].
```

```
add_shortcode( 'ksiazka', 'boj_sc1_book' );
```

```
// Funkcja wywoływana w celu zastąpienia znacznika [ksiazka].
```

```
function boj_sc1_book() {
```

```
    return '<a href="http://www.amazon.com/dp/0470560541">książka<a/>';
```

```
}
```

```
?>
```

---

*Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc1.php`.*

Jak to działa?

1. Za pomocą funkcji `add_shortcode()` zarejestrowano znacznik `[ksiazka]` jako nowy skrót, który będzie zastąpiony przez dane wyjściowe wygenerowane przez funkcję `boj_sc1_book()`.
2. Wywoływana funkcja zastępująca skrót (tutaj: `boj_sc1_book()`) musi na końcu zwrócić wartość za pomocą polecenia `return`. Błędem bardzo często popełnianym przez początkujących jest użycie do wyświetlenia wartości polecenia `echo` zamiast `return`. Trzeba pamiętać, że w tym przypadku nie można użyć polecenia `echo`.

Po aktywacji wtyczki można już napisać: „Kup moją `[ksiazka]`” na stronie wpisu bloga, a znacznik zostanie zastąpiony odnośnikiem prowadzącym na stronę w serwisie Amazon.

Warto zwrócić uwagę, że platforma WordPress jest bardzo elastyczna w zakresie składni skrótu: można stosować znaczniki mniej lub więcej przypominające znaczniki XHTML dowolnego typu, np. `[ksiazka]`, `[ksiazka ]`, `[ksiazka/]` i `[ksiazka /]`. Jedynym warunkiem jest, aby pomiędzy nawiasem otwierającym znacznik i samym znacznikiem nie było spacji.

## [ksiazki tytuł="xkcd"]

Co zrobić w sytuacji, gdy chcesz promować więcej niż jedną książkę?

Pierwszą opcją będzie utworzenie wielu skrótów w sposób przedstawiony powyżej, po jednym dla każdej książki (np. [ksiazka1], [ksiazka2], [ksiazka3] itd.). Jednak znacznie bardziej eleganckim rozwiązaniem będzie użycie atrybutu dla skrótu. W ten sposób można zastosować sprytniejszą składnię w postaci [ksiazki tytul="prowp"] lub [ksiazki tytul="xkcd"].

W kodzie ponownie będzie użyta ta sama funkcja `add_shortcode()`, ale tym razem z nowym parametrem `$attr`, który pobiera tablicę atrybutu — parę wartości.



```
<?php
/*
```

```
Plugin Name: Przykład skrótu nr 2
```

```
Plugin URI: http://przyklad.pl/
```

```
Description: Zastępuje znacznik [ksiazki tytul="xxx"] różnymi odnośnikami prowadzącymi do serwisu
↳ Amazon.
```

```
Version: 1.0
```

```
Author: Ozh
```

```
Author URI: http://wrox.com/
```

```
*/
```

```
// Rejestracja nowego skrótu: [ksiazki tytul="xxx"].
```

```
add_shortcode( 'ksiazki', 'boj_sc2_multiple_books' );
```

```
// Funkcja wywoływana w celu zastąpienia znacznika [ksiazki].
```

```
function boj_sc2_multiple_books( $attr ) {
    switch( $attr['tytul'] ) {
        case 'xkcd':
            $asin = '0615314465';
            $title = 'XKCD Volume 0';
            break;
        default:
            case 'prowp':
                $asin = '0470560541';
                $title = 'Professional WordPress';
                break;
    }
    return "<a href='http://www.amazon.com/dp/$asin'>$title<a/>";
}
?>
```

---

*Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc2.php`.*

Jak to działa?

1. Za pomocą funkcji `add_shortcode()` zarejestrowano znacznik [ksiazki] jako nowy skrót.
2. Funkcja odpowiedzialna za zastąpienie znacznika `boj_sc2_multiple_books()` oczekuje podania parametru `$attr` w postaci tablicy atrybutu — pary wartości do użycia w skrótce. Przykładowo po użyciu znacznika [ksiazki tytul="prowp"] funkcja otrzyma parametr w postaci `array( 'tytul' => 'prowp' )`.
3. Funkcja odpowiedzialna za zastąpienie znacznika będzie zwracała różne wartości w zależności od przekazanego jej atrybutu.
4. Użycie znacznika [ksiazki] bez atrybutu spowoduje, że funkcja otrzyma pusty ciąg tekstowy. W takim przypadku zwrócona będzie wartość domyślna zdefiniowana w kodzie.

## [amazon asin="12345"]tytuł książki[/amazon]

Przygotowaną powyżej wtyczkę można jeszcze bardziej usprawnić, czyli umożliwić sparametryzowanie tekstu w odnośniku prowadzącym do serwisu Amazon.



Użyta będzie ponownie ta sama funkcja `add_shortcode()`, ale tym razem z drugim parametrem — `$content` — który będzie przekazywał ciąg tekstowy wykorzystany następnie jako tekst w odnośniku.



```
<?php
```

```
/*
```

```
Plugin Name: Przykład skrótu nr 3
```

```
Plugin URI: http://przyklad.pl/
```

```
Description: Zastępuje znacznik [amazon isbn="xxx"]tytuł książki[/amazon].
```

```
Version: 1.0
```

```
Author: Ozh
```

```
Author URI: http://wrox.com/
```

```
*/
```

```
// Rejestracja nowego skrótu: [amazon isbn="123"]tytuł książki[/amazon].
```

```
add_shortcode( 'amazon', 'boj_sc3_amazon' );
```

```
// Funkcja wywoływana w celu zastąpienia znacznika [amazon].
```

```
function boj_sc3_amazon( $attr, $content ) {
```

```
    // Pobranie numeru ASIN (Amazon Standard Identification Number).
```

```
    if ( isset( $attr['asin'] ) ) {
```

```
        $asin = preg_replace( '/[^\\d]/', '', $attr['asin'] );
```

```
    } else {
```

```
        $asin = '0470560541';
```

```
    }
```

```
    // Oczyszczenie treści lub ustawienie domyślnej.
```

```
    if ( !empty( $content ) ) {
```

```
        $content = esc_html( $content );
```

```
    } else {
```

```
        if ( $asin == '0470560541' ) {
```

```
            $content = 'Professional WordPress';
```

```
        } else {
```

```
            $content = 'ta książka';
```

```
        }
```

```
    }
```

```
    return "<a href='http://www.amazon.com/dp/$asin'>$content<a/>";
```

```
}
```

```
?>
```

---

*Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc3.php`.*

Jak to działa?

1. Zarejestrowano kolejny skrót, tym razem używając znacznika `[amazon]`.
2. Funkcja odpowiedzialna za zastąpienie znacznika `boj_sc3_amazon()` oczekuje dwóch opcjonalnych parametrów: tablicy atrybutu — pary wartości do użycia w skrócie oraz ciągu tekstowego umieszczonego pomiędzy znacznikami skrótu otwierającym i zamykającym.
3. Funkcja odpowiedzialna za zastąpienie znacznika musi obsłużyć różne kombinacje brakującego atrybutu w postaci numeru ASIN (ang. *Amazon Standard Identification Number*) i (lub) tytułu książki: znaczniki `[amazon]`, `[amazon asin="123"]` i `[amazon]super książka[/amazon]` działają bez problemów.
4. Skrót może zwrócić dowolną treść, więc trzeba pamiętać o zastosowaniu technik omówionych w rozdziale 6. Numer ASIN trzeba oczyścić, aby składał się jedynie z cyfr. Ponadto należy się upewnić, że tytuł książki może być bezpiecznie wyświetlony w Twoim blogu i nie zniszczy znacznika `<a>`, w którym będzie umieszczony.

## Podsumowanie: funkcja `add_shortcode()` i funkcja odpowiedzialna za zastąpienie znacznika

Podczas rejestracji nowego skrótu dwa parametry definiują wzorzec znacznika w nawiasach kwadratowych oraz funkcję wywoływaną w celu zastąpienia znacznika:

```
<?php
add_shortcode( 'boj', 'boj_my_shortcode' );
?>
```

Funkcja odpowiedzialna za zastąpienie znacznika pobiera dwa parametry (puste, jeśli zostaną pominięte), czyli tablicę atrybutu — para wartości — oraz ciąg tekstowy definiujący treść umieszczoną pomiędzy znacznikami skrótu otwierającym i zamykającym. Podobnie jak w kodzie HTML, wielkość liter w atrybutach nie ma znaczenia.

Tak jak w każdej innej funkcji PHP, istnieje możliwość zdefiniowania wartości domyślnych. Trzeba pamiętać, że funkcja odpowiedzialna za zastąpienie znacznika musi zwrócić wartość.

```
<?php
function boj_my_shortcode( $attr = array( 'var' => 'wartość' ), $content =
↳ 'książka' ) {
    // $attr to tablica asocjacyjna.
    // $content to ciąg tekstowy.
    return $something;
}
?>
```

Atrybuty skrótu nie rozróżniają wielkości liter, mogą mieć dowolną wartość lub nie mieć jej wcale, a także obsługują znaki cudzysłowu lub akceptują ich brak. Przedstawione poniżej przykłady pokazują wartości tablicy `$attr` w funkcji odpowiedzialnej za zastąpienie znacznika, w zależności od sposobu użycia skrótu:

- `[boj]` : `$attr` będzie pustym ciągiem tekstowym;
- `[boj hello]` : `$attr` będzie tablicą `array( 'hello' )`;
- `[boj name=ozh skillz='1337' MAP="q3dm6"]` : `$attr` będzie tablicą `array ( 'name' => 'ozh', 'skillz' => '1337', 'map' => 'q3dm6' )`.

## Wskazówki dotyczące skrótów

Skróty to doskonały sposób wzbogacenia wpisu bloga skomplikowaną i dynamiczną treścią. W celu zagwarantowania dostarczenia użytkownikowi najlepszych wrażeń autor wtyczki powinien pamiętać o dwóch podstawowych zasadach:

- zbuduj prostą i niezawodną wtyczkę;
- pamiętaj, że jest dynamiczna.

## Pomyśl o prostocie

Użytkownik jest zadowolony, kiedy otrzymuje nowe funkcje do bloga i jednocześnie może stosować skróty pozwalające na wyświetlanie znacznie bardziej skomplikowanej treści. Jednak zapamiętywanie składni parametrów skrótu jest niewygodne: powstaje wrażenie konieczności poznania nowego języka znaczników.

Powróćmy jeszcze na chwilę do skrótu [amazon]: istnieje możliwość przygotowania wtyczki, która będzie dodawała skrót [amazonobraz] wyświetlający obraz produktu z serwisu Amazon. Zadaniem użytkownika jest podanie numeru ASIN, typu obrazu (książka czy płyta CD) oraz jego wielkości.

Po zbudowaniu wtyczka będzie zezwalała na użycie skrótu, takiego jak [amazonobraz asin='12345' typ='CD' wielkoscobrazu='maly'].

Kiedy użytkownik posiada tę wtyczkę od dłuższego czasu, może zapomnieć nazwy i składni atrybutów. Czy to było [amazonobraz], czy [obrazamazon]? Czy atrybutem jest isbn, czy asin? Czy wielkoscobrazu='ogromny', czy 'duzy'? Czy typ='CD', czy typ='dysk'?

Wprawdzie posiadanie dużej liczby opcji może być zaletą, ale równocześnie nie chcesz, aby użytkownicy musieli nieustannie sięgać do dokumentacji, ponieważ to sprawia niekorzystne wrażenie. Warto sprawę uprościć i pozwolić użytkownikom na instynktowne używanie wtyczki.

Po wprowadzeniu modyfikacji kod wtyczki przedstawia się następująco:



```
<?php
<?php
/*
```

*Plugin Name: Przykład skrótu nr 4*

*Plugin URI: http://przyklad.pl/*

*Description: Zastąpienie znacznika [amazonobraz] obrazem z serwisu Amazon.*

*Version: 1.0*

*Author: Ozh*

*Author URI: http://wrox.com/*

```
*/
```

*// Rejestracja skrótów [amazonobraz] i [obrazamazon]*

```
add_shortcode( 'amazonobraz', 'boj_sc4_amazonimage' );
add_shortcode( 'obrazamazon', 'boj_sc4_amazonimage' );
```

*// Funkcja odpowiedzialna za zastąpienie skrótu [amazonobraz].*

```
function boj_sc4_amazonimage( $attr, $content ) {
```

*// Pobranie numeru ASIN lub ustawienie domyślnego.*

```
$possible = array( 'asin', 'isbn' );
$asin = boj_sc4_find( $possible, $attr, '0470560541' );
```

*// Pobranie powiązanego identyfikatora lub ustawienie domyślnego.*

```
$possible = array( 'aff', 'affiliate' );
$aff = boj_sc4_find( $possible, $attr, 'aff_id' );
```

*// Pobranie wielkości obrazu, jeśli podano.*

```
$possible = array( 'wielkosc', 'obraz', 'wielkoscobrazu' );
$size = boj_sc4_find( $possible, $attr, '' );
```

*// Pobranie typu, jeśli podano.*

```
if( isset( $attr['type'] ) ) {
    $type = strtolower( $attr['typ'] );
    $type = ( $type == 'cd' or $type == 'dysk' ) ? 'cd' : '';
}
```

*// Utworzenie adresu URL prowadzącego do obrazu w serwisie Amazon.*

```
$img = 'http://images.amazon.com/images/P/';
```

```

$img .= $asin;
// Opcje obrazu: wielkość.
if( $size ) {
    switch( $size ) {
        case 'maly':
            $size = '_AA100';
            break;
        default:
        case 'sredni':
            $size = '_AA175';
            break;
        case 'duzy':
        case 'ogromny':
            $size = '_SCLZZZZZZ';
            break; // Dobra praktyka, nie zapomnij o ostatnim poleceniu break.
    }
}
// Opcje obrazu: typ.
if( $type == 'cd' ) {
    $type = '_PF';
}
// Dołączenie opcji do adresu URL, o ile podano jakiegokolwiek opcje.
if( $type or $size ) {
    $img .= '.01.'. $type. $size;
}
// Zakończenie tworzenia adresu URL obrazu.
$img .= '.jpg';

// Trzeba pamiętać o zwróceniu obrazu.
return "<a href='http://www.amazon.com/dp/$asin'><img src='$img' /></a>";
}

// Funkcja pomocnicza:
// Wyszukuje $find_keys w tablicy $in_array, zwraca $default, jeśli nie znajdzie $find_keys.
function boj_sc4_find( $find_keys, $in_array, $default ) {
    foreach( $find_keys as $key ) {
        if( isset( $in_array[$key] ) )
            return $in_array[$key];
    }
    return $default;
}
?>

```

---

*Powyższy fragment kodu pochodzi z pliku plugin\_boj\_sc4.php.*

Najpierw uwagę należy zwrócić na rejestrację dwóch skrótów wywołujących tę samą funkcję: w ten sposób użytkownik może użyć skrótu zarówno [amazonobraz], jak i [obrazamazon].

Następnie warto zwrócić uwagę na to, jak wiele atrybutów jest traktowanych jako synonimy: przy użyciu funkcji pomocniczej o nazwie boj\_sc4\_find() główna funkcja odpowiedzialna za obsługę skrótu sprawdza wartości \$attr['asin'] i \$attr['isbn']. Gdy ich brakuje powoduje ustawienie wartości domyślnej.

Kiedy informacje nie są związane z platformą WordPress, warto także przyjrzeć się sposobowi tworzenia przez wtyczkę odnośnika do obrazu w serwisie Amazon. Podstawowy adres URL to <http://images.amazon.com/images/P/>; do niego są dołączane następujące elementy:

- numer ASIN, np. w postaci B0020EBMN4;
- jeśli mają być użyte opcje, trzeba dołączyć .01.;
- pierwsza użyta opcja to wielkość: dołączenie \_AA100 powoduje pobranie obrazu o szerokości 100 pikseli, podczas gdy dołączenie \_SCLZZZZZZ pobiera dużą wersję obrazu;
- inna możliwa opcja to pobranie obrazu płyty CD: w tym celu do tworzonego adresu URL trzeba dołączyć \_PF;
- na końcu adres URL obrazu trzeba zakończyć, dodając .jpg.

Po aktywacji wtyczki można utworzyć nowy wpis bloga o treści Aktualnie słucham [amazonobraz asin="B00008WT5E" typ="cd" wielkosc= "maly"], a otrzymany wynik będzie podobny do pokazanego na rysunku 10.2.

The screenshot shows the WordPress 'Edytuj wpis' (Edit post) interface. On the left is a sidebar menu with options like 'Wpisy', 'Media', 'Odnosniki', etc. The main area is the post editor for a post titled 'Krótki test'. The content area contains the text 'Aktualnie słucham [amazonobraz asin="B00008WT5E" typ="cd" wielkosc="maly"]'. Below the editor, a status bar shows 'Ścieżka: p', 'Liczba słów: 7', and 'Autorem:'. On the right, a preview of the published post is shown. The preview displays the title 'Krótki test', the date 'Opublikowano 18 września 2011', and the generated image of a CD. An arrow points from the shortcode in the editor to the CD image in the preview. Below the image, the text 'Aktualnie słucham' is visible, along with a category link 'Zaszufladkowano do kategorii Bez kategorii | Napisz odpowiedź'.

**Rysunek 10.2.** Wtyczka pobierająca dane z serwisu Amazon w działaniu

## Pamiętaj o dynamiczności

Dane skrótu są generowane dynamicznie: za każdym razem, gdy platforma WordPress wyświetla stronę (pojedynczy wpis bloga lub archiwum), treść skrótu jest przetwarzana, a wszystkie skróty zastępowane są danymi zwracanymi przez wywołania funkcji obsługujących te skróty.

Zamienniki, takie jak wykorzystane w rozdziale, są bardzo szybkie, więc podczas rejestrowania nowego skrótu nie trzeba się przejmować wydajnością działania platformy WordPress.

Jednak wydajność nabierze większego znaczenia, gdy skróty będą pobierały informacje z bazy danych bądź ze zdalnych witryn internetowych.

- W pierwszym przypadku kod spowoduje wykonanie dodatkowych zapytań SQL, co może drastycznie zmniejszyć wydajność w wolniejszych serwerach.
- W drugim przypadku kod będzie wykonywał zewnętrzne żądania HTTP, które mogą spowolnić wygenerowanie całej strony, bo platforma WordPress będzie oczekiwała na odpowiedź ze zdalnego serwera.

W takich przypadkach warto rozważyć buforowanie wyniku przetworzenia skrótu, np. w meta-danych wpisu bloga. W kolejnej wtyczce zostanie zaimplementowana taka technika buforowania.

## Wewnętrzny sposób działania

Oprócz `add_shortcode()` służącej do rejestracji nowego skrótu istnieją także inne interesujące funkcje. Ponadto warto poznać kilka faktów dotyczących API Shortcode, które można wykorzystać w tworzonych wtyczkach.

### `$shortcode_tags`

Wszystkie zarejestrowane skróty są przechowywane w tablicy globalnej o nazwie `$shortcode_tags` w postaci par `'nazwa_skrótu' => 'funkcja_obsługująca_dany_skrót'`:

```
<?php
global $shortcode_tags;
var_dump( $shortcode_tags );
/* Wynik:
array (
    'wp_caption' => 'img_caption_shortcode',
    'caption' => 'img_caption_shortcode',
    'gallery' => 'gallery_shortcode',
    'embed' => '__return_false',
    'amazonimage' => 'boj_sc4_amazonimage',
    'amazonimg' => 'boj_sc4_amazonimage',
)
*/
?>
```

### `remove_shortcode()`

Przy użyciu funkcji `remove_shortcode()` mamy możliwość dynamicznego wyrejestrowania skrótu, np.:

```
remove_shortcode( 'amazonobraz' );
```

### `remove_all_shortcodes()`

Podobnie, w celu dynamicznego wyrejestrowania wszystkich skrótów należy użyć funkcji `remove_all_shortcodes()` bez parametrów. Z technicznego punktu widzenia wymieniona funkcja po prostu zeruje tablicę globalną `$shortcode_tags`, która staje się pusta.

## strip\_shorcodes()

Funkcja `strip_shorcodes()` powoduje usunięcie zarejestrowanych skrótów z treści, co przedstawiono w poniższym przykładzie:

```
<?php
$content = <<<S
Pewne istniejące skrótóy: [amazonobraz] [gallery]
Nieistniejące skrótóy: [bleh] [123]
S;
echo strip_shortcode( $content );
/* Result:
Pewne istniejące skrótóy:
Nieistniejące skrótóy: [bleh] [123]
*/
?>
```

## shortcode\_atts()

Funkcja może być użyta do porównania atrybutów podanych przez użytkownika względem listy obsługiwanych atrybutów, a następnie do ustawienia wartości domyślnych, jeśli będzie trzeba.

Spójrz np. na sposób działania wbudowanego skrótu `[gallery]`. Odpowiedzialna za jego obsługę funkcja to `gallery_shortcode()`, która przetwarza atrybuty skrótu następująco:

```
<?php
function gallery_shortcode( $attr ){
    // Zdefiniowanie atrybutów obsługiwanych oraz ich wartości domyślne.
    $defaults = array(
        'order'      => 'ASC',
        'orderby'    => 'menu_order ID',
        'id'         => $post->ID,
        'itemtag'    => 'dl',
        'icontag'    => 'dt',
        'captiontag' => 'dd',
        'columns'    => 3,
        'size'       => 'thumbnail',
        'include'    => '',
        'exclude'    => ''
    );
    // Filtrowanie atrybutów podanych przez użytkownika oraz ustawienie ich wartości domyślnych,
    // ↪ jeśli zostały pominięte.
    $options = shortcode_atts( $defaults, $attr );
    // [... dalsza część kodu...]
    // Plik: wp-includes/media.php
}
?>
```

Dość długa lista obsługiwanych atrybutów oraz ich wartości domyślne zdefiniowane w tablicy `$defaults` są łączone z atrybutami dostarczonymi przez użytkownika w tablicy `$attr`. Wszystkie nieznanne atrybuty są ignorowane.

## do\_shortcode()


Funkcja `do_shortcode()` przeszukuje pod kątem skrótów przekazany jej jako parametr ciąg tekstowy treści, a następnie przetwarza je. Podczas inicjalizacji platformy WordPress jest powiązana z filtrem `the_content`, więc zajmuje się treścią wpisu bloga:

```
<?php
// W pliku wp-includes/shortcodes.php
add_filter( 'the_content', 'do_shortcode', 11 );
?>
```

## Skróty rekurencyjne

Może się zdarzyć, że treść skrótu będzie zawierała inne skróty. Przykładowo możesz zarejestrować skróty `[b]` i `[i]` do wyświetlania tekstu pogrubionego i zapisanego kursywą. Skróty powinny działać w zagnieżdżonej strukturze, takiej jak `[b]pewien [i]ciekaw[y]/i] tekst[/b]`.

Struktura taka nie stanowi problemu, ponieważ funkcja obsługująca dany skrót może rekurencyjnie wywołać funkcję `do_shortcode()`:



```
<?php
// Dodanie skrótów [b] i [i].
add_shortcode( 'i', 'boj_sc5_italic' );
add_shortcode( 'b', 'boj_sc5_bold' );
// Funkcja obsługująca skrót: zwraca pogrubiony tekst.
function boj_sc5_bold( $attr, $content ) {
    return '<strong>' . do_shortcode( $content ) . '</strong>';
}
// Funkcja obsługująca skrót: zwraca tekst zapisany kursywą.
function boj_sc5_italic( $attr, $content ) {
    return '<em>' . do_shortcode( $content ) . '</em>';
}
?>
```

---

*Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc5.php`.*

Każda funkcja obsługująca skrót upewnia się, że przetworzony zostanie skrót znajdujący się w tekście danego skrótu.


## Kod BBCode we wtyczce obsługującej komentarze

Na tym etapie można utworzyć nową wtyczkę, która umożliwi stosowanie w komentarzach znaczników typu BBCode. W większości forów zamiast zwykłych znaczników HTML, np. `<a>` lub `<b>`, komentujący muszą stosować znaczniki, takie jak `[url]` i `[b]`.

Wtyczka będzie ponadto charakteryzowała się następującymi cechami.

- Nie będzie zmieniać sposobu tworzenia wpisów bloga przez autorów (czyli z użyciem znaczników HTML).
- W komentarzach nie będą stosowane skróty zarejestrowane do używania w innych miejscach wpisu bloga, takie jak np. zdefiniowany wcześniej skrót `[amazonobraz]` lub `[gallery]`.

Poniżej przedstawiono pełny kod wtyczki.



```
<?php
/*
Plugin Name: Przykład skrótu nr 6
```



```

Plugin URI: http://przyklad.pl/
Description: Umożliwia stosowanie skrótów [url] i [b] w komentarzach.
Version: 1.0
Author: Ozh
Author URI: http://wrox.com/
*/

// Rejestracja dla zaczepu 'comment_text' funkcji odpowiedzialnej za przetworzenie komentarza.
add_filter( 'comment_text', 'boj_sc6_comments' );

// Funkcja przetwarzająca treść komentarza.
function boj_sc6_comments( $comment ) {

    // Zapisanie zarejestrowanych skrótów.
    global $shortcode_tags;
    $original = $shortcode_tags;

    // Wyrejestrowanie wszystkich skrótów.
    remove_all_shortcodes();

    // Zarejestrowanie nowych skrótów.
    add_shortcode( 'url', 'boj_sc6_comments_url' );
    add_shortcode( 'b', 'boj_sc6_comments_bold' );
    add_shortcode( 'strong', 'boj_sc6_comments_bold' );

    // Usunięcie wszystkich znaczników HTML z komentarza.
    $comment = wp_strip_all_tags( $comment );

    // Przetworzenie treści komentarza z uwzględnieniem skrótów.
    $comment = do_shortcode( $comment );

    // Wyrejestrowanie skrótów komentarza, przywrócenie zapisanych skrótów.
    $shortcode_tags = $original;

    // Zwrócenie komentarza.
    return $comment;
}

// Skrót [b] lub [strong] powoduje wywołanie funkcji zastępującej skrót znacznikiem <strong>.
function boj_sc6_comments_bold( $attr, $text ) {
    return '<strong>' .do_shortcode( $text ) . '</strong>';
}

// Skrót [url] powoduje wywołanie funkcji zastępującej skrót znacznikiem <a>.
function boj_sc6_comments_url( $attr, $text ) {
    $text = esc_url( $text );
    return "<a href=\"\$text\">$text</a>";
}
?>

```

---

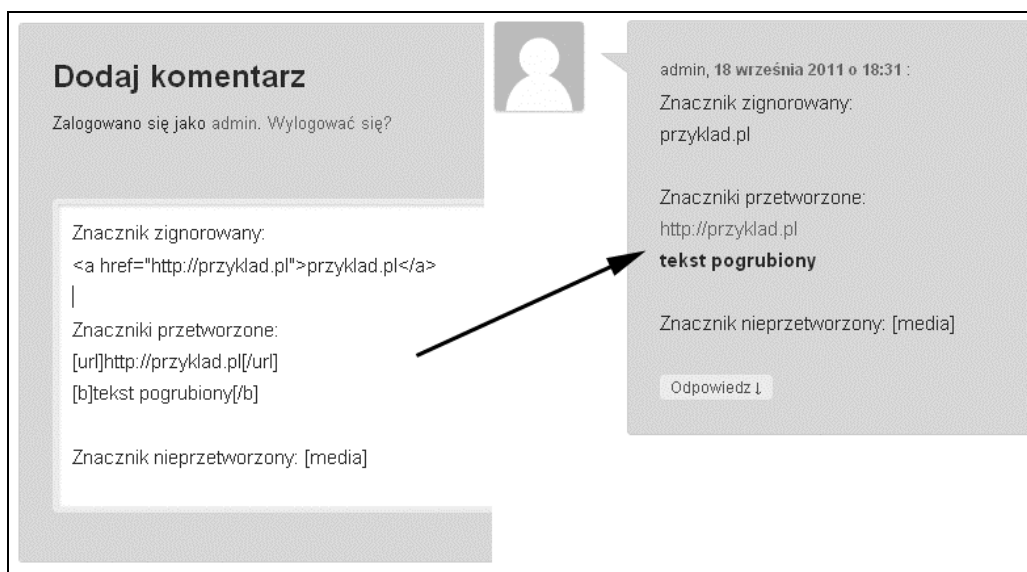
Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc6.php`.

Jak to działa?

1. Jak możesz zobaczyć, wtyczka nie powoduje zarejestrowania nowych skrótów [url] i [b] od razu na początku, ponieważ w takim przypadku wpływałyby one na treść wpisu bloga. Pierwszym zadaniem wtyczki jest przechwycenie treści komentarza.

2. Funkcja przetwarzająca komentarz `boj_sc6_comments()` najpierw tworzy kopię wszystkich skrótów, a następnie je wyrejestrowuje.
3. Kolejnym krokiem jest rejestracja nowych skrótów: `[url]` i `[b]`. (W celu ułatwienia życia użytkownikowi `[strong]` jest odpowiednikiem `[b]`).
4. Z treści komentarza przechowywanej w zmiennej `$comment` następuje usunięcie zwykłych znaczników HTML, a następnie zastosowanie nowo zarejestrowanych skrótów.
5. Warto zwrócić uwagę, jak funkcja odpowiedzialna za pogrubienie tekstu rekurencyjnie wywołuje funkcję `do_shortcode()`, co pozwala na stosowanie zagnieżdżonych struktur.
6. Kolejny krok to przywrócenie oryginalnych skrótów, przy okazji następuje wyrejestrowanie skrótów komentarza `[url]` i `[b]`.
7. Sformatowana treść komentarza zostaje zwrócona w celu wyświetlenia.

Teraz możesz aktywować wtyczkę i dodać nowy komentarz. Spójrz na rysunek 10.3, aby przekonać się, jak ignorowane są znaczniki HTML. Skrót `[b]` i `[url]` są przetwarzane, ale zwykłe skróty, takie jak `[gallery]`, które są przetwarzane we wpisie bloga, są w komentarzach ignorowane.



Rysunek 10.3. Wtyczka umożliwiająca stosowanie znaczników typu BBCode w działaniu

## Ograniczenia skrótów podczas obsługi struktur zagnieżdżonych

Jak się wcześniej dowiedziałeś, platforma WordPress potrafi obsługiwać zagnieżdżone struktury skrótów przy założeniu, że funkcje odpowiedzialne za ich obsługę rekurencyjnie wywołują `do_shortcode()`. Jednak tego rodzaju rozwiązanie ma swoje ograniczenia i czasami może po prostu nie działać, o czym się wkrótce przekonasz.

Struktura przedstawiona poniżej jest obsługiwana prawidłowo, ponieważ zagnieżdżone skróty są inne, a każdy z nich jest prawidłowo osadzony:

Działają:  
[foo]

```

[bar]
  [baz]
[/bar]
[/foo]

```

Struktura przedstawiona niżej nie będzie prawidłowo obsługiwana, bo osadzony skrót jest taki sam jak skrót, w którym się zawiera:

Nie działa:

```

[foo]
  [foo]
  [/foo]
[/foo]

```

Pamiętaj, że skróty mogą być samozamykające się (samodzielny skrót [foo] lub [foo/]) bądź mogą zawierać treść ([foo]treść[/foo]). Wspomniana treść również może uniemożliwiać prawidłowe przetwarzanie pewnych struktur:

Nie działa:

```

[foo]
[foo]
  treść
[/foo]

```

## Integracja z usługą Google Mapy

Jako pełny i znacznie bardziej skomplikowany przykład użycia skrótów utworzymy wtyczkę pozwalającą na integrację usługi Google Mapy z Twoją witryną bazującą na platformie WordPress.

Google oferuje wiele różnych API pozwalających na uzyskanie dostępu do usług, szczególnie do usługi map. Usługa ta bazuje na dwóch powiązanych z nią usługach: API Google Geocoding i API Google Maps.



### UWAGA

Google udostępnia dokładną dokumentację API Google Maps. Więcej informacji na ten temat można znaleźć na stronie <http://code.google.com/apis/maps/documentation/javascript/>.

W tym podrozdziale utworzymy wtyczkę pozwalającą na konwersję adresu zapisanego w postaci zwykłego tekstu (np. ul. Kościuszki 1c, 44-100 Gliwice) na dynamicznie generowaną, interaktywną mapę Google.

## Uzyskanie dostępu do API Google Geocoding

Pierwszym krokiem podczas konwersji adresu na mapę jest utworzenie „geokodu” adresu. Tworzenie geokodu to proces opisany jako konwersja standardowego adresu (np. ul. Kościuszki 1c, 44-100 Gliwice) na współrzędne geograficzne (50.289, 18.660). Współrzędne są używane przez API Google Maps do znalezienia wskazanego położenia na mapie oraz umieszczenia na niej znaczników zależnych od podanych współrzędnych.

Obecnie API Google Geocoding zwraca wyniki w dwóch formatach: JSON i XML. W omawianym tutaj przykładzie zostanie wykorzystany format JSON. Ponadto będziemy stosować techniki omówione w rozdziale 9., który jest poświęcony wykonywaniu i obsłudze żądań HTTP.

Firma Google uprościła proces współpracy z API. W celu otrzymania współrzędnych można wykonać żądanie na następujący adres URL: [http://maps.google.com/maps/api/geocode/\\$output? \\$parameters](http://maps.google.com/maps/api/geocode/$output?$parameters), gdzie \$output oznacza format danych wyjściowych (np. 'json'), natomiast \$parameter to ciąg tekstowy zapytania zawierający parametry dodatkowe dla geokodu.

Do API trzeba przekazać tylko dwa wymagane parametry: address lub latLng i sensor.

- Ponieważ nie są znane współrzędne geograficzne, użyty będzie parametr address. Ten parametr to zapisany w postaci zwykłego tekstu pełny adres, który ma zostać zamieniony na geokod. Adres jest zakodowany w adresie URL.
- Parametr sensor wskazuje, czy żądanie pochodzi z urządzenia zawierającego sensor lokalizacji (np. smartfon). Tej zmiennej ustawimy wartość false.

Możesz to bardzo łatwo przetestować: wczytaj API Google Geocoding poprzez uruchomienie dowolnej przeglądarki internetowej i przejdź na stronę <http://maps.google.com/maps/api/geocode/json?address=Kościuszki+1c+Gliwice&sensor=false>.

Jak widać, zwrócone dane JSON zawierają współrzędne geograficzne podanego adresu, a także dane dodatkowe, takie jak kod pocztowy (który nie został podany w żądaniu).

Teraz możemy przystąpić do tworzenia funkcji `boj_gmap_geocode()`, która wygeneruje geokod dla wskazanego adresu:

```
<?php
// Utworzenie geokodu na podstawie adresu: wartością zwrótną jest tablica zawierająca współrzędne
↳geograficzne.
function boj_gmap_geocode( $address ) {
    // Utworzenie adresu URL do API Google Geocoding.
    $map_url = 'http://maps.google.com/maps/api/geocode/json?address=';
    $map_url .= urlencode( $address ).' & sensor=false';
    // Wykonanie żądania GET.
    $request = wp_remote_get( $map_url );
    // Pobranie obiektu JSON.
    $json = wp_remote_retrieve_body( $request );
    // Upewnienie się, że żądanie zakończyło się powodzeniem. W przeciwnym razie trzeba zwrócić false.
    if( empty( $json ) )
        return false;
    // Odkodowanie obiektu JSON.
    $json = json_decode( $json );
    // Pobranie współrzędnych geograficznych.
    $lat = $json->results[0]->geometry->location->lat; // Szerokość geograficzna.
    $long = $json->results[0]->geometry->location->lng; // Długość geograficzna.
    // Zwrócenie tablicy zawierającej współrzędne geograficzne.
    return compact( 'lat', 'long' );
}
?>
```

Funkcja wykonuje żądanie do API Google Geocoding i otrzymuje odpowiedź w formacie JSON, która po zdekodowaniu będzie zawierała współrzędne geograficzne. Można to sprawdzić, analizując otrzymaną wartość zwrótną:

```
<?php
$coords = boj_gmap_geocode( 'ul. Kościuszki 1c, Gliwice' );
var_dump( $coords );
/* Wynik:
array(2) {
```

```

["lat"]=> float(50.28891549)
["long"]=> float(18.65953990)
}
*/
?>

```

Więcej informacji wraz z objaśnieniem funkcji użytych w powyższej funkcji można znaleźć w rozdziale 9.

## Przechowywanie wyników

Jednym z ważniejszych aspektów skrótów jest fakt, że za każdym razem dynamicznie generują swoją treść. Jednak wykonywanie żądania HTTP do API Google Geocoding podczas każdego wyświetlenia wpisu bloga nie będzie efektywnym rozwiązaniem, ponieważ doprowadzi do spowolnienia wczytywania każdej strony.

Alternatywą jest przechowywanie współrzędnych danego adresu w metadanych dołączanych do wpisu bloga. W ten sposób podczas kolejnego wyświetlenia wpisu bloga współrzędne geograficzne zostaną pobrane z bazy danych wraz z pozostałymi metadanymi wpisu bloga. W ten sposób unika się wykonywania dodatkowego żądania HTTP.



### UWAGA

Metadane wpisu bloga dostępne w interfejsie WordPress są pobierane razem z samym wpisem bloga. Tak więc odczyt tych informacji nie powoduje wykonania dodatkowego zapytania SQL. Więcej informacji na temat metadanych znajduje się w rozdziale 11.

Zamiast pobierać współrzędne z API Google za pomocą funkcji `boj_gmap_geocode()`, można użyć funkcji proxy o nazwie `boj_gmap_get_coords()`, która sprawdza dostępność tych informacji w metadanych wpisu bloga. Jeżeli informacje nie zostaną znalezione w metadanych, będą pobrane z API Google, a następnie zapisane w metadanych w celu ich późniejszego wykorzystania.

Poniżej przedstawiono kod funkcji proxy:

```

<?php
// Konwersja adresu zapisanego w postaci zwykłego tekstu na współrzędne geograficzne.
// Współrzędne będą pobrane z metadanych, o ile to możliwe. W przeciwnym razie będą pobrane z API Google.
function boj_gmap_get_coords( $address = 'ul. Kościuszki 1c, Gliwice' ) {
    // Identyfikator bieżącego wpisu bloga.
    global $id;
    // Sprawdzenie, czy współrzędne geograficzne znajdują się w bazie danych.
    $saved = get_post_meta( $id, 'boj_gmap_addresses' );
    foreach( (array)$saved as $_saved ) {
        if( isset( $_saved['address'] ) && $_saved['address'] == $address ) {
            extract( $_saved );
            return compact( 'lat', 'long' );
        }
    }
    // Współrzędne nie są jeszcze buforowane, więc trzeba je pobrać z Google.
    $coords = boj_gmap_geocode( $address );
    if( !$coords )
        return false;
}

```

```
// Buforowanie wyniku w metadanych wpisu bloga.
add_post_meta( $id, 'boj_gmap_addresses', array(
    'address' => $address,
    'lat' => $coords['lat'],
    'long' => $coords['long']
)
);
extract( $coords );
return compact( 'lat', 'long' );
}
?>
```

Gdy adres będzie po raz pierwszy konwertowany na geokod, wywołanie `add_post_meta()` spowoduje wstawienie do metadanych (o nazwie `boj_gmap_addresses`) wpisu bloga tablicy, podobnej do przedstawionej poniżej:

```
array(
    "address" => "ul. Kościuszki 1c, Gliwice",
    "lat"      => "50.28891549",
    "long"     => "18.65953990"
)
```

Podczas kolejnego wczytywania strony współrzędne geograficzne powinny już być znalezione i pobrane wraz z metadanymi wpisu bloga.

## Uzyskanie dostępu do API Google Maps

Kiedy wiadomo, w jaki sposób skonwertować adres na współrzędne geograficzne przy użyciu API Google Geocoding, można przystąpić do wykorzystania tych współrzędnych w usłudze Google Mapy za pomocą API Google Maps.

### Koncepcje API

Interaktywne mapy usługi Google Mapy są tworzone za pomocą kodu JavaScript. Ten kod musi być wstawiony na stronie, na której ma zostać wyświetlona mapa. Przed rozpoczęciem integracji usługi z wtyczką warto dowiedzieć się, jak można osadzić mapę na stronie HTML.

Na początek trzeba dodać skrypt główny:

```
<script type="text/javascript" src="http://maps.google.com/maps/api/js?sensor=false">
```

Następnie należy wstawić kod JavaScript dotyczący map i mieszczący się we własnej funkcji:

```
function initialize_map() {
```

Obiekt przechowuje nowy egzemplarz Google Maps wraz z podanymi parametrami szerokości i długości geograficznej:

```
var myLatLng = new google.maps.LatLng(45.124099, -123.113634);
```

Inny obiekt może zdefiniować opcje mapy: poziom przybliżenia, wycentrowanie oraz rodzaj mapy (teren, mapa, satelita lub widok hybrydowy):

```
var myOptions = {
    zoom: 4,
    center: myLatLng,
    mapTypeId: google.maps.MapTypeId.SATELLITE
}
```

Teraz mapę można już dołączyć do obiektu HTML, takiego jak `<div>`; w omawianym przykładzie obiekt ma atrybut o identyfikatorze `map_canvas`:

```
var map = new google.maps.Map( document.getElementById("map_canvas"), myOptions );
```

Przedstawiony poniżej ciąg tekstowy przechowuje własny tekst wyświetlający informacje w oknie, które się pokazuje po kliknięciu znacznika znajdującego się na mapie:

```
var contentString = '<div id="content">'+
  '<p><b>Ikona Firefox</b>: Gdzieś w Oregonie, ta ikona o średnicy 67 '+
  'metrów została utworzona przez grupę użytkowników systemu Linux Uniwersytetu '+
  'Stanowego w celu świętowania wydania przeglądarki Firefox 2</p>'+
  '</div>'+
  '</div>';
```

Ten ciąg tekstowy jest teraz dołączony do nowego egzemplarza obiektu `InfoWindow`:

```
var infowindow = new google.maps.InfoWindow({
  content: contentString
});
```

Pozostało umieszczenie znacznika na mapie:

```
var marker = new google.maps.Marker({
  position: myLatLng,
  map: map,
  title: 'Ikona Firefox'
});
```

Trzeba jeszcze zdefiniować odpowiednie zachowanie, czyli wyświetlenie okna z informacjami po kliknięciu znacznika:

```
google.maps.event.addListener(marker, 'click', function() {
  infowindow.open(map,marker);
});
}
```

Prawie gotowe! Teraz tworzymy pusty obiekt HTML, który otrzyma mapę, i wywołujemy funkcję JavaScript odpowiedzialną za wyświetlenie mapy:



```
<p>Mapa jest wyświetlona poniżej:</p>
<div id="map_canvas" style="width:600px;height:600px"></div>
<script type="text/javascript">initialize_map(</script>
```

*Powyższy fragment kodu pochodzi z pliku `google_map_api_example.html`.*

W pliku przedstawiono koncepcję używania API Google Maps w celu wyświetlenia mapy (zobacz rysunek 10.4).

Teraz jesteś przygotowany do implementacji dynamicznej mapy Google we własnej wtyczce.

## Implementacja wtyczki

Trzeba przygotować główną część wtyczki: funkcję rejestrującą skrót `[googlemap]` i wyświetlającą mapę Google. Skrót będzie używany w następujący sposób:

```
[googlemap width=500 height=300 zoom=12]ul. Kościuszki 1c, Gliwice[/googlemap]
```



**Rysunek 10.4.** Gotowa mapa wyświetlona w przeglądarce internetowej

Najpierw należy zarejestrować sam skrót. Można również zarejestrować podobne skróty, np. [googlemaps], [google\_map] i [google\_maps], które będą obsługiwane przez tę samą funkcję.

```
<?php
// Dodanie obsługi skrótu [googlemap].
add_shortcode( 'googlemap', 'boj_gmap_generate_map' );
```

Kolejny krok to rozpoczęcie definiowania funkcji odpowiedzialnej za analizę i przetworzenie atrybutów skrótu oraz jego treści:

```
// Funkcja odpowiedzialna za obsługę skrótu.
function boj_gmap_generate_map( $attr, $address ) {
    // Ustawienie wartości domyślnych mapy.
    $defaults = array(
        'width' => '500',
        'height' => '500',
        'zoom' => 12,
    );
    // Pobranie atrybutów mapy (w przypadku ich pominięcia będą miały wartości domyślne).
    extract( shortcode_atts( $defaults, $attr ) );
```

W części pierwszej następuje przygotowanie tablicy wartości domyślnych połączonych z rzeczywistymi atrybutami przy użyciu funkcji `shortcode_atts()`, która zwraca tablicę. Wywołanie `extract()` powoduje zaimportowanie zmiennych z tablicy w taki sposób, że dla egzemplarza `array( 'size' => 300 )` otrzymamy `$size = 300`.

```
// Pobranie współrzędnych.
$coord = boj_gmap_get_coords( $address );
// Upewniamy się, że mamy współrzędne. W przeciwnym razie trzeba zwrócić pusty ciąg tekstowy.
if( !$coord )
    return '';
```



W tym miejscu utworzono geokod na podstawie adresu (albo pobrano z API albo z bazy danych wraz z metadanymi wpisu bloga). Gdy utworzenie geokodu zakończy się niepowodzeniem (np. na skutek tymczasowego problemu między serwerami Twoim i Google), należy zwrócić pusty ciąg tekstowy.

```
// Dane wyjściowe skrótu.
$output = '';
// Przypisanie wartości zmiennym $lat i $long.
extract( $coord );
```

Po przygotowaniu wszystkich wymaganych zmiennych, a przed wygenerowaniem danych wyjściowych, trzeba je oczyścić. Niektóre będą umieszczone w ciągach tekstowych JavaScript, jeszcze inne użyte jako atrybuty elementów HTML. Konieczne jest więc prawidłowe ich oczyszczenie, zgodnie z opisem przedstawionym w rozdziale 6.

```
// Oczyszczenie zmiennych w zależności od kontekstu, w którym będą użyte.
$lat      = esc_js( $lat );
$long     = esc_js( $long );
$address  = esc_js( $address );
$zoom     = esc_js( $zoom );
$width    = esc_attr( $width );
$height   = esc_attr( $height );
```

Teraz przechodzimy do kodu JavaScript.

Główny skrypt najczęściej umieszczany jest w elemencie <head> dokumentu HTML, ale w tym przypadku to nie jest najlepsze rozwiązanie, ponieważ skrypt będzie wówczas wstawiony nawet na tej stronie, na której nie jest potrzebny.

Lepiej skrypt osadzić w kodzie jako fragment kodu skrótu. W ten sposób znajdzie się tylko na wymagającej go stronie.

```
// Wygenerowanie unikalnego identyfikatora mapy, co pozwala mieć wiele map na stronie.
$map_id = 'boj_map_' . md5( $address );
// Tylko jednokrotne dodanie głównego skryptu Google Maps na stronie.
static $script_added = false;
if( $script_added == false ) {
    $output .= '<script type="text/javascript"
        src="http://maps.google.com/maps/api/js?sensor=false" ></script>';
    $script_added = true;
}
```

Teraz można wstawić kod JavaScript odpowiedzialny za obsługę mapy. Każda funkcja i każde miejsce zarezerwowane dla mapy będą miały unikalne nazwy, uzyskamy to, używając wcześniej wygenerowanej zmiennej \$map\_id, co pozwala na umieszczenie wielu map na tej samej stronie.

```
// Dodanie kodu dotyczącego mapy.
$output .= <<<CODE
<div id="$map_id" ></div>
<script type="text/javascript" >
    function generate_map_id() {
        var latlng = new google.maps.LatLng( $lat, $long );
        var options = {
            zoom: $zoom,
            center: latlng,
            mapTypeId: google.maps.MapTypeId.ROADMAP
        }
    }
```

```

var map = new google.maps.Map(
    document.getElementById("$map_id"),
    options
);
var legend = '<div class="map_legend"><p> $address </p></div> ';
var infowindow = new google.maps.InfoWindow({
    content: legend,
});
var marker = new google.maps.Marker({
    position: latlng,
    map: map,
});
google.maps.event.addListener(marker, 'click', function() {
    infowindow.open(map,marker);
});
}
generate_$map_id();
</script>

```

Do danych wyjściowych dodajemy proste style w postaci atrybutów zdefiniowanych przez użytkownika:

```

<style type="text/css">
.map_legend{
width:200px;
max-height:200px;
min-height:100px;
}
#$map_id {
width: {$width}px;
height: {$height}px;
}
</style>

```

CODE;

Oczywiście, nie wolno zapomnieć o zwróceniu treści zamiennika dla skrótu:



```

return $output;
}
?>

```

---

*Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc7.php`.*

Na tym etapie wtyczka jest już gotowa do użycia! Utwórz nowy wpis bloga i zastosuj np. poniższy skrót:

```
[googlemap width=450 height=300 zoom=14]ul. Kościuszki 1c, Gliwice[/googlemap]
```

Tak utworzony wpis bloga powinien być podobny do pokazanego na rysunku 10.5.

## Więcej pomysłów dotyczących skrótów

Przy użyciu skrótów można bardzo łatwo dodawać interesujące i praktyczne funkcje, tutaj ograniczeniem jest jedynie Twoja wyobraźnia. Istnieje np. możliwość zarejestrowania skrótu wyświetlającego treść tylko dla zalogowanych użytkowników, wyświetlającego treść zależną ograniczoną czasowo, zaciemniającego adres e-mail oraz wiele innych. W kolejnych punktach zostaną przedstawione niektóre opcje.

Edytuj wpis Dodaj nowy

## Integracja z usługą Google Mapy

Bezpośredni odnośnik: <http://localhost/wordpress/?p=32> Zmocz

Opublikowano 18 września 2011

Wyslij/wstaw

Spotkajmy się tutaj!

[gogoglemap width=450 height=300 zoom=14]ul. Kości

Spotkajmy się tutaj!

Ścieżka: p

Liczba słów: 10 Autorem ostatniej edycji jest admin,

Rysunek 10.5. Wpis bloga zintegrowany z mapą Google

## Wyświetlanie treści jedynie dla zalogowanych użytkowników

Pierwszy krótki implementowany skrót to elegancki sposób wyświetlenia treści jedynie zalogowanym użytkownikom. Jego sposób użycia we wpisie bloga może być następujący:

Dzisiejsza porada Jedi:

[czlonkowie]Używaj mocy[/czlonkowie]

Funkcja odpowiedzialna za obsługę wspomnianego skrótu znajduje się w poniższym fragmencie kodu:

```

<?php
add_shortcode( 'czlonkowie', 'boj_sc8_loggedin' );
function boj_sc8_loggedin( $attr, $content ) {
    if( is_user_logged_in() ) {
        return $content;
    } else {
        return "<p>Tylko dla zalogowanych użytkowników</p>";
    }
}
?>

```

Powyższy fragment kodu pochodzi z pliku `plugin_boj_sc8.php`.

Wynikiem jest treść umieszczona pomiędzy znacznikami skrótu [czlonkowie]. Będzie ona wyświetlona tylko zalogowanym użytkownikom. Przykład tej samej koncepcji, jednak omówiony bardziej szczegółowo, znajduje się w rozdziale 8.

## Wyświetlenie treści ograniczonej czasowo

Innym prostym i przydatnym skrótem jest kod pozwalający na wyświetlenie treści ograniczonej czasowo, np. odnośnik promocyjny ważny przez jedynie 24 godziny:

Ten odnośnik promocyjny jest ważny przez jedynie 24 godziny:  
[24godziny] <http://przyklad.pl/promo/> [/24godziny]

W celu zaimplementowania takiego skrótu trzeba wykorzystać przedstawiony poniżej fragment kodu, który po prostu sprawdza bieżącą godzinę oraz godzinę opublikowania wpisu bloga:



```
<?php
add_shortcode( '24godziny', 'boj_sc8_24hours' );
function boj_sc8_24hours( $attr, $content ) {
    $now = time();
    $post_time = get_the_date( 'U' );
    if( ( $now - $post_time ) > 86400 ) {
        return 'Oferta wygasła!';
    } else {
        return $content;
    }
}
?>
```

---

*Powyższy fragment kodu pochodzi z pliku plugin\_boj\_sc8.php.*

Jeżeli bieżący wpis bloga był opublikowany wcześniej niż 86 400 sekund wcześniej (czyli 24 godziny), to tekst znajdujący się między znacznikami [24godziny] nie zostanie wyświetlony.

## Zaciemnienie adresu e-mail

W kolejnym krótkim i użytecznym skrócie pokazano praktyczny sposób konwersji adresu e-mail zapisanego w postaci zwykłego tekstu na odnośnik `mailto:`, który będzie zaciemniony (tzn. mniej czytelny) dla robotów spamowych. We wpisie bloga wystarczy po prostu użyć skrótu:

Napisz do mnie na adres [email]ozh@ozh.org[/email]

Funkcja obsługująca skrót wykorzystuje funkcję WordPress o nazwie `antispambot()`, która po prostu konwertuje znaki na encje HTML, które są trudniejsze do odczytania przez roboty spamowe i roboty kolekcjonujące adresy e-mail:



```
<?php
add_shortcode( 'email', 'boj_sc8_email' );
function boj_sc8_email( $attr, $content ) {
    if( is_email( $content ) ) {
        $content = antispambot( $content );
        return sprintf( '<a href="mailto:%s" > %s <a/>', $content, $content );
    }
    else {
        return '';
    }
}
?>
```

---

*Powyższy fragment kodu pochodzi z pliku plugin\_boj\_sc8.php.*

Umieszczenie adresu e-mail między znacznikami [email][/*email*] spowoduje zaciemnienie tego adresu przez skrót. Przykładowo po podaniu adresu *ozh@ozh.org* wartością zwrotną skrótu będzie `&#122;&#104;&#64;&#111;z&#104;&#46;org`.

## Podsumowanie

Skróty otwierają użytkownikom końcowym drzwi do zaawansowanej, dostosowanej do własnych potrzeb i dynamicznej treści. Ci użytkownicy nie muszą znać skomplikowanego kodu HTML, JavaScript, CSS lub PHP albo mogą nie mieć ochoty na jego tworzenie.

Za pomocą skrótów można przygotować dla klientów zaawansowane makra i znacznie zwiększyć wartość swojej pracy przy niewielkim wysiłku.



# Skorowidz

- #wordpress, 502
- #wordpress-dev, 502
- \$args
  - blog\_id, 211
  - can\_export, 310
  - capabilities, 308
  - capability\_type, 308
  - exclude, 211
  - exclude\_from\_search, 307
  - has\_archive, 308
  - hierarchical, 308
  - include, 211
  - labels, 308
  - menu\_icon, 309
  - menu\_position, 309
  - meta\_compare, 211
  - meta\_key, 211
  - meta\_value, 211
  - number, 212
  - offset, 212
  - order, 211
  - orderby, 211
  - permalink\_epmask, 310
  - public, 307
  - publicly\_queryable, 307
  - query\_var, 309
  - register\_meta\_box\_cb, 310
  - rewrite, 309
  - role, 211
  - search, 211
  - show\_in\_nav\_menus, 309
  - show\_ui, 307
  - supports, 307
  - taxonomies, 309
- \$shortcode\_tags, 288
- \$userdata
  - admin\_color, 215
  - comment\_shortcuts, 215
  - description, 215
  - display\_name, 215

- first\_name, 215
- ID, 214
- last\_name, 215
- nickname, 215
- rich\_editing, 215
- role, 215
- user\_email, 215
- user\_login, 215
- user\_nicename, 215
- user\_pass, 214
- user\_registered, 215
- user\_url, 215
- %1\$s, 125, 127
- %2s, 127
- %, 125, 126
- \_\_return\_empty\_array, 68
- \_\_return\_false, 68
- \_\_return\_true, 68
- \_\_return\_zero, 68

## A

- admin, 210
- Administracja siecią, 452
- adres URL, 40–41, 155, 401
- agent użytkownika, 258
- agregacja treści sieci, 429
- Ajax, *Patrz* technologia Ajax
- akcje, 51, 52
- akcje Ajax, 364
- Akismet, 26
- aktualizacja oprogramowania, 458
- aktywacja wtyczki, 29, 42, 450
- album\_artist, 336
- album\_genre, 336
- All in One SEO Pack, 26
- alternatywne API, 275
- amazon, 281
- analizator składni RSS, 276
- Apache, 402

- API Google Geocoding, 293, 294
- API Google Maps, 293, 296
  - implementacja wtyczki, 298
  - osadzanie mapy, 296
  - rejestracja skrótu, 298
- API Google QR Code, 417
- API HTTP, 247, 249, 252, 277
- API Options, 175
  - parametr autoloadd, 179, 180
  - rozdzielanie opcji wtyczki, 179
  - usuwanie opcji, 178
  - wczytywanie tablicy opcji, 178
  - zapisywanie opcji, 175
- API Rewrite, 401, 419, 421
- API Settings, 391
  - dodawanie pól, 188, 189
  - dodawanie sekcji, 189
  - funkcje, 181
  - generowanie formularza, 185
  - rejestracja ustawień, 182
  - sekcje i ustawienia, 183
  - strona administracyjna, 181
  - tworzenie kodu HTML, 185
  - weryfikacja błędów, 187
  - weryfikacja danych, 184
  - zarządzanie wtyczką, 185
- API Shortcode, 279
- API Transients
  - pobieranie opcji, 192
  - usunięcie opcji, 193
  - zapisywanie opcji, 192
- apostrof, 47
- argument
  - can\_export, 310
  - capabilities, 308, 313, 327
    - assign\_terms, 327
    - delete\_terms, 327
    - edit\_terms, 327
    - manage\_terms, 327
  - capability\_type, 308, 313

## argument

- exclude\_from\_search, 307
- has\_archive, 308
- hierarchical, 308, 325
- labels, 308, 326
  - klucze tablicy, 326
- menu\_icon, 309
- menu\_position, 309
- permalink\_epmask, 310
- post\_type, 317
- public, 307, 325
- publicly\_queryable, 307
- query\_var, 309, 325
- register\_meta\_box\_cb, 310
- rewrite, 309, 325
  - hierarchical, 325
  - slug, 325
  - with\_front, 325
- show\_in\_nav\_menus, 309, 326
- show\_tagcloud, 325
- show\_ui, 307, 325
- supports, 307
- taxonomies, 309
- update\_count\_callback, 325

## arkusze stylów, 351

## ASIN, 283

## atak typu

- CSRF, 137
- SQL Injection, 162, 170
- XSS, 144

## atak związany z nadużyciem

- uprawnień, 134

## attribut class="more-link", 365

**B**

## baza danych, 24, 455

## baza danych MySQL, 506

## baza danych zaczepów, 500

## BBCode, 279, 290

## bezpieczeństwo, 371

## bezpieczeństwo wtyczki, 133

- dobrze nawyki, 172

## biała lista, 161

## biblioteka jQuery, 337, 349

## bit.ly, 137

## blog, 306

- dodawanie metadanych, 320
- metadane wpisu, 319
- możliwe typy wpisów, 306

## możliwości, 313

- pętla własnego typu, 315
- pobieranie metadanych, 321
- pobieranie treści, 317
- przypisanie taksonomii, 329
- rejestracja własnego typu
  - wpisu, 306, 310
- taksonomia, 314
- typ wpisu, 314, 318
- uaktualnienie metadanych
  - wpisu, 322
- ustawianie etykiet, 311
- usuwanie metadanych, 322
- usuwanie wersji wpisu, 385
- własne typy wpisów, 306, 315
- wpisy, 330

## błąd 404, 262

## błąd serwera, 262

## błędy, 373, 461

- rejestracja, 466
- tryb usuwania, 461, 462
- włączanie rejestracji, 466
- wyłączanie wyświetlania, 467
- wyświetlanie komunikatów, 462

## body, 256

## BOJ Admin Lang, 194

## branding wtyczki, 485

## Brown Adam, 500

## buforowanie danych, 467

## buforowanie danych we wtyczce, 470

**C**

## Chrome, 374

## ciągi tekstowe, 312

## CMS, 194

## Coda, 505

## Codex, 497

- opis funkcji, 498

- przeszukiwanie witryny, 497

## Contact Form 7, 26

## cookies, 256

## cron, 375, 399

- harmonogram zadań, 376

- metody wykonywania zadań,

## 385

## Cross Site Scripting, 133

## cudzysłów, 47

## CURL, 251, 260

## czat WordPress, 502

**D**

## dane JSON, 264, 294

## dane krótkotrwałe, 193, 194

## dane skrótu, 287

## dane użytkownika, 219

- display\_name, 219

- ID, 219

- user\_email, 219

- user\_login, 219

- user\_nicename, 219

- user\_pass, 219

- user\_registered, 219

- user\_url, 219

## dane wyjściowe, 415, 428

Dashboard Widgets, *Patrz*

- widżety kokpitu, 24, 93

Database, *Patrz* baza danych

## DATEDIFF, 396

## definiowanie sekcji, 183

## definiowanie ustawień, 183

## delete\_users, 237

## demon cron, 375

## description, 84

## dezaktywacja wtyczki, 29, 43

## deinstalacja wtyczki, 44

## DNS, 262

## dodanie możliwości do roli, 241

## dodawanie

- elementu menu, 80

- menu, 77

- podmenu, 78

- pól, 189, 191, 197

- sekcji, 189, 190

- skryptów, 353

- strony w oparciu o szablon, 412

- treści, 118

- własnego pola użytkownika, 97

## dokumentacja, 493

- kodu, 46

- osadzona w kodzie, 494

- WordPress, 497

## dołączanie skryptów, 345

## DOM, 341

## dowolne zapytania, 169

## dynamiczność, 287

## dynamiczny skrypt, 358

## dyrektywa mod\_rewrite, 403

## dziennik błędów, 467



**E**

edytor tekstu, 495, 505  
 edytor wtyczek, 31  
 e-mail, 302  
 etykiety, 311  
   add\_new, 312  
   add\_new\_item, 312  
   edit\_item, 312  
   name, 312  
   new\_item, 312  
   not\_found, 312  
   not\_found\_in\_trash, 312  
   parent\_item\_colon, 313  
   search\_items, 312  
   singular\_name, 312  
   view\_item, 312

**F**

filtr, 51, 61  
 filtr plugins\_api, 274  
 filtrowanie odpowiedzi, 258  
 filtrowanie zaawansowane, 260  
 filtrowanie znaczników, 148  
 filtrowanie żądań, 258  
 Firebug, 158, 374, 504  
 Firefox, 374, 504  
 fora pomocy technicznej, 501  
 formularz opcji widgetu, 439  
 FSF, 474  
 FTP, 505  
 fundacja FSF, 474  
 funkcja  
   \_\_(), 118  
   \_\_return\_empty\_array(), 67  
   \_\_return\_false(), 261  
   \_e(), 118  
   \_ex(), 120  
   \_n(), 122  
   \_n\_noop(), 123  
   \_nx(), 123  
   \_nx\_noop(), 124  
   \_x(), 120  
 abs(), 145  
 absint(), 147, 158, 439  
 add\_action(), 71, 100, 496  
 add\_blog\_option(), 448  
 add\_comments\_page(), 82  
 add\_dashboard\_page(), 81

add\_feed(), 419  
 add\_filter(), 71, 496  
 add\_links\_page(), 82  
 add\_management\_page(), 82  
 add\_media\_page(), 82  
 add\_menu\_page(), 77  
 add\_meta\_box(), 97  
 add\_option(), 179  
 add\_options\_page(), 82  
 add\_pages\_page(), 82  
 add\_permastruct(), 413  
 add\_plugins\_page(), 82  
 add\_post\_meta(), 320  
 add\_posts\_page(), 81  
 add\_rewrite\_rule(), 407  
 add\_rewrite\_tag(), 413  
 add\_role(), 239  
 add\_settings\_error(), 187  
 add\_settings\_field(), 183  
 add\_settings\_section(), 183  
 add\_shortcode(), 281, 284, 432  
 add\_submenu\_page(), 79  
 add\_theme\_page(), 82  
 add\_user\_meta(), 195, 223  
 add\_users\_page(), 82  
 add\_user\_to\_blog(), 449, 450  
 admin\_url(), 364  
 aktywacji wtyczki, 42  
 apply\_filters(), 61, 72, 75  
 apply\_filters\_ref\_array(), 63, 72, 75  
 array\_map(), 158  
 author\_can(), 234  
 boj\_addfeed\_add\_feed(), 420  
 boj\_altapi\_check(), 273  
 boj\_cron\_pester\_check(), 390  
 boj\_cron\_rev\_delete(), 386  
 boj\_cron\_rev\_setting\_input(), 387  
 boj\_display\_user\_website(), 219  
 boj\_ep\_display\_json(), 417  
 boj\_ep\_display\_qr(), 417  
 boj\_gmap\_geocode(), 294  
 boj\_gmap\_get\_coords(), 295  
 boj\_install(), 43  
 boj\_list\_users\_of\_blog(), 213  
 boj\_mbe\_create(), 97  
 boj\_mbe\_function(), 97  
 boj\_multisite\_widget(), 437  
 boj\_multisite\_switch\_page(), 429  
 boj\_myplugin\_validate\_options(), 184  
 boj\_sc6\_comments(), 292  
 boj\_styling\_settings(), 108  
 boj\_ti\_ask\_twitter(), 266  
 boj\_ti\_get\_infos(), 266  
 boj\_utags\_do\_action(), 142  
 boj\_view\_cron\_settings(), 382  
 check\_admin\_referer(), 139  
 check\_ajax\_referer(), 372  
 checked(), 89, 392  
 compare(), 395  
 count\_many\_users\_posts(), 222  
 count\_user\_posts(), 221  
 count\_users(), 213  
 create\_function(), 67  
 current\_filter(), 66  
 current\_time(), 496  
 current\_user\_can(), 134, 136, 232, 237  
 current\_user\_can\_for\_blog(), 233  
 dbDelta(), 204  
 deactivate\_plugins(), 43  
 delete\_blog\_option(), 448  
 delete\_option(), 178  
 delete\_post\_meta(), 322  
 delete\_user\_meta(), 196, 226  
 dezaktywacji, 43  
 did\_action(), 57  
 do\_action(), 72, 75  
 do\_action\_ref\_array(), 54, 72, 75  
 do\_settings\_fields(), 189  
 do\_settings\_sections(), 189  
 do\_shortcode(), 290, 292  
 error\_log(), 260  
 error\_reporting(), 145  
 esc\_attr(), 155, 438  
 esc\_attr\_\_(), 118  
 esc\_attr\_e(), 119  
 esc\_attr\_x(), 121  
 esc\_html(), 155  
 esc\_html\_\_(), 119  
 esc\_html\_e(), 119  
 esc\_html\_x(), 121  
 esc\_js(), 156  
 esc\_sql(), 162

- funkcja
- esc\_url(), 156
  - esc\_url\_raw(), 95, 103
  - fetch\_feed(), 276
  - fopen(), 250
  - fsockopen(), 251
  - function\_exists(), 458
  - get\_avatar(), 496
  - get\_blog\_count(), 454
  - get\_blog\_details(), 428, 438
  - get\_blog\_option(), 448
  - get\_blog\_post(), 428
  - get\_currentuserinfo(), 135, 220, 496
  - get\_num\_queries(), 172
  - get\_option(), 177, 387
  - get\_pages(), 496
  - get\_post(), 229
  - get\_post\_meta(), 99
  - get\_post\_types(), 497
  - get\_posts(), 496
  - get\_query\_var(), 409
  - get\_role(), 241
  - get\_sitstats(), 454
  - get\_super\_admins(), 453
  - get\_taxonomy(), 330
  - get\_the\_author\_description(), 465
  - get\_user\_count(), 454
  - get\_user\_meta(), 224
  - get\_userdata(), 219, 496
  - get\_users(), 211, 212
  - get\_users\_of\_blog(), 212
  - grant\_super\_admin(), 452
  - have\_posts(), 430
  - has\_action(), 56
  - has\_filter(), 65
  - home\_url(), 41
  - insert(), 164
  - intval(), 145, 147
  - is\_blog\_user(), 449
  - is\_email(), 151, 495
  - is\_int(), 147
  - is\_multisite(), 427, 430
  - is\_super\_admin(), 237
  - is\_tax(), 334
  - is\_taxonomy(), 462
  - is\_taxonomy\_hierarchical(), 333
  - is\_user\_logged\_in(), 210
  - is\_wp\_error(), 255
  - isset(), 100, 465
  - json\_decode(), 264
  - json\_encode(), 264, 417
  - load\_plugin\_textdomain(), 117
  - load\_tweets(), 342
  - map\_meta\_cap(), 232, 235
  - mysql\_fetch\_array(), 163
  - mysql\_query(), 163
  - parse\_request(), 404, 405
  - plugin\_dir\_path(), 40
  - plugins\_api(), 274
  - plugins\_url(), 41
  - post\_type\_exists(), 318
  - prepare(), 386
  - print\_r(), 405
  - printf(), 125
  - register\_deactivation\_hook(), 43
  - register\_activation\_hook(), 496
  - register\_activation\_hook(), 42
  - register\_deactivation\_hook(), 496
  - register\_post\_type(), 307, 313, 314, 319, 497
  - register\_setting(), 182, 184
  - register\_taxonomy(), 324, 327
  - register\_taxonomy\_for\_object\_type(), 329
  - register\_uninstall\_hook(), 46, 496
  - rejestracji błędów, 466
  - remove\_action(), 55
  - remove\_all\_actions(), 56
  - remove\_all\_filters(), 65
  - remove\_all\_shortcodes(), 288
  - remove\_filter(), 64
  - remove\_role(), 240
  - remove\_shortcode(), 288
  - remove\_user\_from\_blog(), 451
  - restore\_current\_blog(), 429
  - revoke\_super\_admin(), 452
  - sanitize\_email(), 151
  - sanitize\_key(), 149
  - sanitize\_text\_field(), 148
  - selected(), 89
  - shortcode\_atts(), 289, 298
  - site\_url(), 41
  - sprintf(), 125
  - strip\_shortcodes(), 289
  - strip\_tags(), 85
  - strtotime(), 150, 390
  - strumienia fopen(), 250
  - switch\_to\_blog(), 429, 431, 440
  - taxonomy\_exists(), 333, 462
  - the\_content(), 317
  - the\_excerpt(), 318
  - the\_permalink(), 318
  - the\_post(), 430
  - the\_terms(), 331
  - the\_title(), 317
  - update(), 84, 164, 439
  - update\_blog\_option(), 448
  - update\_blog\_status(), 447
  - update\_option(), 179
  - update\_post\_meta(), 103, 322
  - update\_user\_meta(), 225
  - user\_can(), 234
  - usort(), 435
  - var\_dump(), 167, 171
  - version\_compare(), 43
  - widget(), 85
  - WordPress Object Cache, 468
  - wp\_add\_dashboard\_widget(), 92
  - wp\_cache\_add(), 469
  - wp\_cache\_delete(), 469
  - wp\_cache\_get(), 469, 470
  - wp\_cache\_replace(), 469
  - wp\_cache\_set(), 469–471
  - wp\_create\_user(), 216
  - wp\_default\_styles(), 351
  - wp\_delete\_user(), 218
  - wp\_dequeue\_script(), 349
  - wp\_enqueue\_script(), 104, 128, 345, 348, 374
    - podstawowy skrypt, 346
    - skrypt w stopce dokumentu, 348
    - skrypt z numerem wersji, 347
    - skrypt z zależnościami, 347
    - własny skrypt, 346
  - wp\_enqueue\_style(), 351
  - wp\_enqueue\_styles(), 104
  - wp\_get\_current\_user(), 220
  - wp\_head(), 60
  - wp\_insert\_user(), 214, 217
  - wpkses(), 157

wp\_localize\_script(), 127, 359  
 wp\_mail(), 28, 496  
 wp\_nav\_menu(), 431  
 wp\_next\_scheduled(), 383  
 wp\_nonce\_field(), 139  
 wp\_nonce\_url(), 138  
 wp\_rand(), 496  
 wp\_redirect(), 496  
 wp\_register\_script(), 351  
 wp\_remote\_get(), 252  
 wp\_remote\_head(), 252  
 wp\_remote\_post(), 252  
 wp\_remote\_request(), 252  
 wp\_schedule\_event(), 376  
 wp\_schedule\_single\_event(), 378  
 wp\_strip\_all\_tags(), 148, 495  
 wp\_unschedule\_event(), 380  
 wp\_upload\_dir(), 496  
 wp\_widget\_rss\_output(), 94  
 wpdb\_query(), 386  
 wpmu\_create\_blog(), 443, 445

**funkcje**

- \*\_option(), 496
- \*\_post\_meta(), 497
- API HTTP, 277
- API Settings, 181
- esc\_\*(), 495
- fabryczne, 47
- obsługi HTTP, 252
- plugin\_dir\_\*(), 496
- szybko zwracające wartość, 67
- uznane za przestarzałe, 459
- WordPress, 495
- wp\_\*\_post(), 496
- wp\_nonce\_\*(), 496
- wp\_remote\_\*
  - parametry domyślne, 253
  - parametry opcjonalne, 253
- zaczepu akcji, 54
- zaczepu filtru, 63

## G

generator dokumentacji, 498  
 generowanie formularza, 185  
 GET, 248, 249  
 Google CDN, 350  
 Google Chrome, 504  
 Google XML Sitemaps, 26

## H

Hackers mailing list, 501  
 harmonogram
 

- jednorazowe zadanie, 378
- stare wersje wpisu bloga, 385
- tworzenie zadań, 376
- usunięcie zadania, 380
- własne odstępy czasu, 381
- wyświetlanie zadań, 381

 headers, 256  
 heredoc, 260  
 HTML, 152  
 HTTP, 24, 247
 

- kody stanu, 249

 http\_request\_failed, 254

## I

i18n, 115  
 identyfikator bloga, 438  
 ikony, 107, 108  
 instalacja wtyczki, 30  
 integracja wtyczki z platformą, 77  
 interfejs użytkownika, 191  
 interfejs użytkownika platformy
 

- WordPress, 106

 internacjonalizacja, 115  
 internacjonalizacja kodu
 

- JavaScript, 127

 is\_wp\_error(), 262  
 iteracja, 343, 365, 382, 383, 404

## J

JavaScript, 156, 337  
 jądro platformy, 493  
 jednorazowe zadanie, 378  
 język domyślny WordPress, 130  
 jQuery, 337
 

- łączenie, 338
- obiekt, 338
- składnia, 338
- tryb bezkonfliktowy, 340
- wykonanie kodu, 340

 JSON, 264

## K

kanal IRC, 502  
 kanal RSS, 95  
 kanal wiadomości, 419
 

- rejestracja, 419
- wtyczka, 420

 katalog
 

- blogs.dir, 426
- boj-alert-box, 127
- boj-alert-box.php, 127
- boj-plugin, 117
- FTP, 261
- languages, 117
- mu-plugins, 450
- plugins, 36
- wp-admin/js, 349
- wp-content, 40, 358, 426, 466
- wp-includes, 495
- wp-includes/js, 349
- wtyczek, 26, 32, 36, 117

 klasa
 

- button-highlighted, 109
- button-primary, 109
- button-secondary, 109
- displaying-num, 113
- form-table, 110
- page-numbers, 113
- SimplePie, 277
- widefat, 112
- WP\_Ajax\_Response, 372
- WP\_Http, 252
- WP\_Widget, 437
- wpdb, 163

 klient Subversion, 480  
 klient wysyłający żądanie, 248  
 klient-serwer, 248  
 kod, 46  
 kod BBCode, 290  
 kod błędu, 254  
 kod błędu 404, 405  
 kod flash, 416  
 kod funkcji proxy, 295  
 kod JavaScript, 296  
 kod kreskowy, 416  
 kod makro, 279  
 kod QR, 406, 416  
 kod źródłowy, 21  
 kod źródłowy dowolnego pliku, 499

kody stanu HTTP, 249  
 kolejkovanie skryptów, 346, 365  
 komentarz, 494  
 komunikaty błędów, 254, 263,  
 373, 462, 464  
 konfiguracja procesu cron, 385  
 konfiguracja SVN, 480  
 konto na witrynie  
 WordPress.org, 478  
 kontrola nad możliwościami,  
 313  
 konwersja cudzysłowu, 144  
 konwersja znaków na encje  
 HTML, 154

## L

l10n, 116  
 licencja, 474  
 Apache, 474  
 BSD, 474  
 dla wtyczki, 39, 474  
 GPL, 474, 475  
 zalety, 475  
 LGPL, 474  
 MIT (X11), 474  
 podwójna, 475  
 podzielona, 475  
 liczba użyta jednokrotnie, 137  
 lista nieuporządkowana, 74  
 lista uporządkowana, 74  
 listy dyskusyjne, 501  
 listy zaczepów, 76  
 lokalne ścieżki dostępu, 40

## M

marketing, 473, 491  
 menu najwyższego poziomu, 429  
 metadane, 194  
 pobieranie, 196  
 uaktualnianie, 195  
 usuwanie, 196  
 zapisywanie, 195  
 metadane użytkownika, 223  
 metadane wpisu bloga, 295, 319  
 metoda  
 GET, 139  
 get\_results(), 168  
 get\_row(), 166

get\_var(), 166  
 insert(), 166, 170  
 prepare(), 170, 171  
 query(), 169  
 remove\_cap(), 242  
 miejsca zarezerwowane, 125  
 model DOM, 341, 353  
 moduł mod\_rewrite, 402  
 modyfikacja pliku .htaccess, 426  
 motyw, 28, 29  
 możliwości, 229, 230, 313  
 możliwości domyślne, 313  
 możliwości meta, 232  
 możliwości roli, 232  
 możliwość  
 delete\_post, 314  
 delete\_forum\_topics, 241  
 delete\_users, 237  
 edit\_others\_posts, 314  
 edit\_post, 314  
 edit\_posts, 314  
 publish\_forum\_topics, 241  
 publish\_posts, 314  
 read\_post, 314  
 read\_private\_content, 238  
 read\_private\_posts, 314  
 music\_album, 335

## N

nagłówek wtyczki, 38  
 nagłówki, 107  
 narzędzia programisty, 493, 504  
 narzędzia służące do  
 tłumaczenia, 130  
 narzędzie  
 GlotPress, 130  
 GNU Gettext, 130  
 KBabel, 130  
 Launchpad, 130  
 phpMyAdmin, 205  
 PHPXref, 499  
 Poedit, 130, 131  
 Pootle, 130  
 nawiasy, 48  
 nazwa wtyczki, 483  
 nazwy domyślne tabel, 202  
 nazywanie  
 funkcji, 47  
 katalogów, 36

plików, 47  
 wtyczek, 36  
 zmiennych, 47  
 NetBeans IDE, 505  
 NextGEN Gallery, 26  
 niebezpieczne dane, 144  
 niebezpieczny kod HTML, 157  
 nieprawidłowy formularz, 143  
 nonce, 137  
 Notepad++, 505  
 number used once, 137  
 numer ASIN, 283

## O

obiekt  
 \$post, 99  
 \$wp\_query, 404, 406  
 \$wp\_rewrite, 404, 419  
 \$wpdb, 163  
 query, 405  
 rewrite, 405  
 wpdb, 172  
 obiekty błędów, 254  
 obraz płyty CD, 287  
 obsługa  
 komentarzy, 290  
 proxy, 257  
 struktur zagnieżdżonych, 292  
 zapytań, 404  
 obiekt query, 405  
 obiekt rewrite, 405  
 proces ogólny, 404  
 wtyczki, 406  
 oczyszczanie danych, 143, 146  
 odnośnik bezpośredni, 412  
 odnośniki, 110, 362, 364  
 odpowiedź serwera WWW  
 body, 256  
 cookies, 256  
 headers, 256  
 response, 256  
 odpowiedź XML, 371, 373  
 ogłoszenie wydania wtyczki, 487  
 ograniczanie dostępu, 231  
 nadawanie uprawnień, 237  
 uprawnienia użytkownika,  
 232  
 opakowanie, 340  
 opakowaniem noConflict(), 341

- opcja
- classname, 83
  - description, 84
  - UTF-8, 131
  - WP\_DEBUG, 466
  - WP\_DEBUG\_LOG, 466
- opcje. 24
- pobieranie, 177, 192
  - rozdzielanie, 179
  - usuwanie, 178, 193
  - wczytywanie tablicy, 178
  - zapisywanie, 175, 192
  - zapisywanie tablicy, 176
- opcje widgetu, 93
- opcje witryny, 448
- open source, 474
- opis funkcji, 498
- opłaty, 476
- Options, *Patrz* opcje
- osadzanie mapy, 296
- osadzanie skryptu, 351
- Ozh Richard, 15
- ## P
- parametr
- \$abs\_rel\_path, 117
  - \$accepted\_args, 53, 55, 62, 64
  - \$action, 138, 372
  - \$after, 317, 331
  - \$arg, 52, 54
  - \$args, 211, 233, 235, 253, 307, 324
  - \$attr, 282
  - \$autoload, 179
  - \$before, 317, 331
  - \$blog\_id, 428, 429, 443, 447, 448, 449, 451
  - \$cap, 235
  - \$capabilities, 239
  - \$capability, 232, 234, 235
  - \$caps, 235
  - \$content, 283
  - \$context, 120
  - \$data, 468
  - \$dependencies, 345
  - \$die, 372
  - \$display\_name, 239
  - \$domain, 117, 443
  - \$echo, 317
  - \$email, 216
  - \$expire, 469
  - \$fields, 428
  - \$file, 40, 42, 44, 45
  - \$function, 42, 44, 45, 53
  - \$function\_to\_check, 57, 66
  - \$function\_to\_remove, 55, 64
  - \$getall, 428
  - \$group, 469
  - \$handle, 127, 345
  - \$id, 218, 331
  - \$in\_footer, 345
  - \$key, 448, 468
  - \$l10n, 127, 129
  - \$meta, 443
  - \$meta\_key, 195, 224, 225, 226, 320, 321, 322, 323
  - \$meta\_value, 195, 224, 225, 226, 320, 322, 323
  - \$more\_link\_text, 318
  - \$object\_name, 127
  - \$object\_type, 324, 330
  - \$output\_type, 166
  - \$password, 216
  - \$path, 41, 443
  - \$plugin, 41
  - \$plugin\_rel\_path, 117
  - \$post, 234
  - \$post\_id, 320, 321, 322, 323, 428
  - \$post\_type, 124, 307
  - \$pref, 447
  - \$prev\_value, 322
  - \$priority, 53, 55, 56, 61, 64, 65
  - \$query-arg, 372
  - \$reassign, 218, 451
  - \$refresh, 448
  - \$role, 239, 449
  - \$row\_offset, 166
  - \$sep, 62, 331
  - \$seplocation, 62
  - \$single, 224, 321
  - \$sql, 166
  - \$src, 345
  - \$stripteaser, 318
  - \$tag, 52, 53, 54, 55, 56, 57
  - \$taxonomy, 324, 329, 331
  - \$text, 125
  - \$title, 62, 443
  - \$unique, 195, 224, 320
  - \$url, 138, 252
  - \$user\_id, 195, 224, 225, 226, 235, 237, 443, 449, 451
  - \$user\_id, 221
  - \$userdata, 214
  - \$username, 216
  - \$validate, 429
  - \$value, 61, 447, 448
  - \$ver, 345
  - \$wp\_title, 62
  - address, 294
  - args, 376, 378
  - autoload, 179, 180
  - blogid, 433
  - body, 254
  - boj\_action, 138
  - boj\_single\_cron\_hook, 379
  - callback, 92, 97
  - callback\_args, 97
  - capability, 78, 79, 80
  - context, 97
  - control\_callback, 92
  - cookies, 254
  - function, 78, 79, 80
  - headers, 254
  - hook, 376, 378
  - icon\_url, 78
  - id, 97
  - key, 99
  - latlng i sensor, 294
  - menu\_slug, 78, 79, 80
  - menu\_title, 78, 79, 80
  - meta\_key, 100
  - meta\_value, 100
  - method, 253
  - num, 433
  - page, 97
  - page\_title, 78, 79, 80
  - parent\_slug, 79
  - pobieranie opcji, 177
  - position, 78
  - post\_id, 99, 100
  - posts\_per\_page, 433
  - prev\_value, 100, 225
  - priority, 97
  - recurrence, 376
  - sensor, 294
  - single, 99
  - store\_id, 407
  - timeout, 253

- parametr
  - timestamp, 376, 378
  - title, 97
  - update-check, 275
  - user-agent, 253
  - widget\_id, 92
  - widget\_name, 92
  - wp\_ajax\_\$action, 362
  - wp\_ajax\_nopriv\_\$action, 362
  - zapisywanie tablicy opcji, 176
- pętla własnego typu, 315
- PHP, 36
- PHPDoc, 47, 494, 498
- PHPDocumentor, 47
- phpMyAdmin, 506
- PHPXref, 47, 498
- Planet WordPress, 503
- plik
  - boj-alert-box-\$locale.mo, 130
  - .htaccess, 402
  - admin-ajax.php, 364
  - boj-alert-box-\$locale.po, 130
  - boj-alert-box-script.js, 129
  - boj-error-plugin.php, 463
  - boj-force-admin-color.php, 217
  - boj-insert-user.php, 215
  - boj-meta-box.php, 104
  - boj-meta-image.js, 105
  - boj-meta-image.js, 103
  - boj-music-collection.php, 335
  - boj-user-favorite-post.php, 228
  - boj-user-ratings.php, 221
  - boj-user-registration-date.php, 220
  - debug.log, 466
  - dziennika błędów, 467
  - formatting.php, 495
  - functions.php, 495
  - http.log, 260
  - index.php, 404
  - MO, 130
  - page.php, 409
  - pluggable.php, 496
  - plugin.php, 496, 500
  - PO, 130
  - post.log, 160
  - post.php, 496
  - POT, 131
  - readme.txt, 480, 482
    - Changelog, 483
    - Contributors, 482
    - Description, 482
    - Donate link, 482
    - Extra, 483
    - Frequently Asked Questions, 483
    - Installation, 482
    - Requires at least, 482
    - Screenshots, 483
    - Stable tag, 482
    - Tags, 482
    - Tested up to, 482
  - script.js.php?args, 359
  - self\_form.php, 158
  - sklepy.php, 409
    - kod źródłowy strony, 410
  - skrypt.js.php, 357
  - testsql.php, 205
  - uninstall.php, 37, 44, 45
  - wp-admin/includes/
    - deprecated.php, 460
  - wp-admin/includes/ms.php, 453
  - wp-config.php, 33, 130, 203, 358, 385
  - wp-includes/class-http.php, 252
  - wp-includes/compat.php, 264
  - wp-includes/deprecated.php, 460
  - wp-includes/110n.php, 199
  - wp-includes/ms-deprecated.php, 460
  - wp-includes/pluggable.php, 135
  - wp-includes/pluggable-deprecated.php, 460
  - wp-includes/post.php, 315
  - wp-includes/post-template.php, 315
  - wp-includes/rewrite.php, 416
  - wp-load.php, 358, 404
  - wp-settings.php, 404
- pliki
  - cookies, 158
  - CSS, 351
  - pl\_PL, 200
  - statyczne.js, 357
  - tłumaczenia, 130
  - tworzące jądro platformy, 495
  - wtyczki boj-music-collection-post-types.php, 310
- Plugin, *Patrz* wtyczka
- pobieranie opcji, 177
- pobranie kolumny, 168
- pobranie ogólnych wyników, 168
- pobranie rekordu, 166
- pobranie zmiennej, 166
- podstrony, 406
- pojemnik zaczepów filtrów, 70
- poła formularza, 110
- poła ustawień, 191
- pole użytkownika, 97
  - inicjalizacja, 99
  - pobranie wartości metadanych, 99
  - pole zaawansowane, 101
  - wstawianie adresu URL, 104
  - wstawianie obrazu, 103
  - zapis danych, 98
  - zapis metadanych, 98
- polecenia SELECT \* FROM, 169
- polecenia SQL, 49
- polecenie
  - crontab, 385
  - echo, 281
  - return, 281
  - testowe SQL, 206
  - wget, 385
- pomoc techniczna, 488
- poprawianie błędów, 463
- POST, 120, 249, 267
- poufne dane, 135
- prefiks, 36
- proces tworzenia wtyczki, 50
- proces wczytywania strony, 25
- promowanie wtyczki
  - fora, 491
  - odnośnik do własnej strony, 491
  - ogłoszenie wydania, 487
  - serwisy społecznościowe, 491
  - tworzenie doskonałego kodu, 486
  - utworzenie własnej strony, 487
  - zbudowanie witryny internetowej, 485

protokół  
 FTP, 506  
 HTTP, 247  
 HTTPS, 364  
 SFTP, 506  
 typu żądanie-odpowiedź, 248  
 proxy, 257  
 przechowywanie danych  
 dostęp do tabeli, 206  
 sprawdzenie istnienia tabeli,  
 203  
 standardowe tabele, 202  
 typy danych, 202  
 uaktualnienie struktury tabeli,  
 204  
 własna tabela, 202  
 przechowywanie współrzędnych  
 adresu, 295  
 przeglądarka Chrome, 374  
 przeglądarka Firefox, 374  
 przeglądarka internetowa, 504  
 przyciski, 109  
 punkt końcowy, 415  
 definicja, 416  
 dodawanie punktu, 416  
 lista stałych, 417  
 PuTTY, 506

## R

read\_private\_content, 238  
 register\_activation\_hook, 58  
 register\_deactivation\_hook, 58  
 reguła zmiany adresów, 407  
 rejestracja ustawień, 182  
 rejestracja widgetu, 87  
 rejestracja zdarzeń, 260  
 rejestrowanie błędów, 466  
 rejestrowanie skryptów, 351  
 repozytorium, 261  
 repozytorium Subversion, 480  
 repozytorium wtyczki, 270, 477  
 informacje szczegółowe, 275  
 uaktualnienie wtyczki, 271  
 zalety, 477  
 response, 256  
 rewrite, 24  
 robot spamowy, 302  
 rola, 229, 230  
 administrator, 242  
 contributor, 241  
 dodanie możliwości, 241

dostosowanie, 238  
 Superadministradora, 452  
 tworzenie, 239  
 usuwanie, 240  
 usuwanie możliwości, 242  
 rola domyślne, 230  
 Administrator, 230  
 Author, 231  
 Contributor, 231  
 Editor, 231  
 Subscriber, 231  
 rola własne, 231  
 administrator forum, 231, 243  
 budowanie wtyczki, 242  
 członek forum, 231, 243  
 moderator forum, 231, 243  
 uprawnienia  
 delete\_forum\_topics, 242  
 edit\_others\_forum\_topics,  
 242  
 publish\_forum\_topics, 242  
 read\_forum\_topics, 242  
 zawieszony członek forum,  
 231, 243  
 rozślawienie wtyczki, 483  
 rozszerzenie Firebug, 504  
 rozszerzenie YSlow, 504  
 RSS, 95, 276

## S

sekcja  
 discussion, 191  
 general, 191  
 media, 191  
 permalink, 191  
 privacy, 191  
 reading, 191  
 writing, 191  
 sekcja WordPress, 191  
 serwer  
 Apache, 402  
 IIS, 402  
 Lighttpd, 402  
 Nginx, 402  
 odpowiedź, 248  
 WWW, 402  
 serwis  
 Amazon, 281  
 Tumblr, 269  
 Twitter, 263  
 Settings, *Patrz* ustawienia  
 SFTP, 505  
 Shortcode, *Patrz* skrót  
 sieć Multisite, 423, 424, 456  
 agregacja treści sieci, 429  
 baza danych, 455  
 funkcje, 427  
 identyfikator bloga, 427  
 instalacja, 425  
 nowa witryna, 443  
 opcje witryny, 448  
 przełączanie witryn, 433  
 sieć i witryna, 424  
 statystyka, 454  
 superadministrator, 452  
 tabele, 455  
 uaktualnianie stanu witryny,  
 447  
 użytkownicy, 448  
 widget, 436  
 właściciel witryny, 453  
 włączanie obsługi, 426  
 zalety, 425  
 SimplePie, 276  
 Site Request Forgery, 133  
 sklep, 407  
 tworzenie strony, 409  
 wyświetlanie produktów, 414  
 składnia \$(), 341  
 składnia dla HTML5, 353  
 składnia dla XHTML, 353  
 skrót, 24, 279, 303  
 skrót do wpisu bloga, 432  
 skrót samozamykający się, 293  
 skróty rekurencyjne, 290  
 skrypt  
 dodawanie, 353  
 dodawanie na stronach  
 administracyjnych, 355  
 dodawanie na stronach  
 publicznych, 356  
 dołączanie, 345  
 osadzanie, 351  
 rejestrowanie, 351  
 strona klienta, 365  
 strona serwera, 366  
 usuwanie, 349  
 wstawianie, 363  
 zastępowanie, 350  
 skrypty Ajax, 142

skrypty domyślne, 349  
 skrypty dynamiczne, 357  
 słowa kluczowe, 131  
 słownik, 497  
 słowo kluczowe KEY, 204  
 SMTP, 25  
 solidna implementacja, 201  
 Source View, 499  
 spacja, 49  
 spotkania  
   WordCamp, 504  
   WordPress Meetup, 504  
 sprawdzanie typu protokołu, 103  
 SQL Injection, 133  
 SSH, 505, 506  
 SSL, 42  
 stała  
   \_\_FILE\_\_, 40  
   FORCE\_SSL\_ADMIN, 364  
   SAVEQUERIES, 135  
   WP\_ACCESSIBLE\_HOSTS, 258  
   WP\_DEBUG, 145  
   WP\_HTTP\_BLOCK\_EXTERNAL, 258  
   WP\_PLUGIN\_URL, 42  
   WP\_UNINSTALL\_PLUGIN, 45  
 standard PHPDoc, 494  
 stopka dokumentu, 348  
 stronicowanie, 113  
 struktura katalogów, 37  
 styl poleceń SQL, 204  
 styl thickbox, 104  
 style, 351  
 style dla znaczników HTML, 112  
 Subversion, 480  
 synchronizacja pliku POT, 131

## Ś

ścieżka dostępu, 39

## T

tabela

  wp\_1\_commentmeta, 455  
 wp\_1\_comments, 455  
 wp\_1\_links, 455  
 wp\_1\_options, 455

  wp\_1\_postmeta, 455  
 wp\_1\_posts, 455  
 wp\_1\_term\_relationships, 456  
 wp\_1\_term\_taxonomy, 456  
 wp\_1\_terms, 456  
 wp\_blog\_versions, 455  
 wp\_blogs, 455  
 wp\_registration\_log, 455  
 wp\_signups, 455  
 wp\_site, 455  
 wp\_sitecategories, 455  
 wp\_sitemeta, 455  
 wp\_usermeta, 455  
 wp\_users, 455

tabela \$wpdb->postmeta, 319

tabela \$wpdb->usermeta, 223

tabela users, 219

tabela wp\_blogs, 428

tabele, 112

  dostęp do tabeli, 206  
 sprawdzenie istnienia, 203  
 standardowe, 202  
 tworzenie tabel, 202  
 uaktualnianie struktury, 204

tablica

  \$\_COOKIE, 158  
 \$\_GET, 145, 158  
 \$\_POST, 145, 158  
 \$\_REQUEST, 145, 158  
 \$results, 169  
 formatów, 165  
 globalna \$shortcode\_tags, 288  
 klauzul WHERE, 164  
 opcji, 84  
 superglobalna \$\_SERVER, 157

tablice danych, 158

Tadlock Justin, 15

tagi warunkowe taksonomii, 333

taksonomia, 314, 323

  kategoria, 323  
 menu nawigacyjne, 323  
 odnośnik, 323  
 pobieranie, 330  
 przypisanie do typu wpisu, 329  
 rejestracja, 324  
 tag wpisu, 323  
 tagi warunkowe, 333

taksonomia hierarchiczna, 327

taksonomia niehierarchiczna, 327

taksonomie własne, 330

technologia Ajax, 337, 341, 343, 360, 374

  dostęp do treści, 344

  informowanie użytkownika, 344

  kod JavaScript, 344

  odpowiedź, 361

  reguły, 360

  usuwanie błędów, 373

  wskaźnik postępu, 344

  zdarzenia, 361

  żądania, 361, 362

testowanie funkcji wtyczek, 33

TextMate, 505

thickbox, 104

tłumaczenie

  dostępne narzędzia, 130

  GlotPress, 130

  GNU Gettext, 130

  KBabel, 130

  Launchpad, 130

  Poedit, 130

  Pootle, 130

tłumaczenie wtyczki, 116, 117

TortoiseSVN, 480

Trac, 501

transakcje HTTP, 248

transients, 24

treści dla zalogowanych, 301

treści ograniczone czasowo, 302

tryb bezkonfliktowy, 340

tryb usuwania błędów, 461

Tumblr, 269

Twitter, 263, 503

Twitter Info, 264

tworzenie

  alternatywnego API, 275

  geokodu, 293

  harmonogramu zadań, 376

  kodu, 46

  koszmarnego kodu, 463

  menu najwyższego poziomu, 77

  nazwy wtyczki, 483

  plików tłumaczenia, 130

  roli, 239

  skrótów, 279

  strony administracyjnej, 181



strony sklepów, 409  
 unikalnych identyfikatorów,  
   137  
 użytkowników, 214  
 widgetów, 82  
 widgetów kokpitu, 92  
 witryn w sieci Multisite, 444  
 własnej tabeli, 202  
 własnych taksonomii, 323  
 własnych zaczepów, 72  
 wtyczki, 35  
 wtyczki z metadanymi  
   użytkownika, 227  
 zaawansowanego widgetu, 90  
 zadań, 376  
 typ wpisu bloga, 314  
 typy danych, 202  
 typy wtyczek, 32  
 typy zwracanej wartości, 67

## U

uaktualnianie  
   komentarzy, 170  
   metadanych, 195  
   struktury tabeli, 204  
 uaktualnienia  
   platformy, 51  
   użytkowników, 214  
   wtyczek, 29  
 udostępnienie wtyczki, 477  
 unieszkodliwianie  
   niebezpiecznych znaków, 121  
 unikalna wartość, 372  
 unikalny identyfikator, 137  
   83a08fcbc2, 137  
   dla adresu URL, 138  
   dla formularza, 139  
   w skryptach Ajax, 142  
 UNIX, 375, 390  
 uprawnienia użytkownika, 134,  
   136  
 uruchamianie sieci, 426  
 usługa skracania adresów URL,  
   137  
 ustawienia, 24  
 ustawienia platformy, 192  
 ustawienia użytkowników, 194  
   dodanie opcji, 198  
   dodawanie pól, 197

  metadane, 194  
   pobieranie identyfikatora, 197  
   pobieranie metadanych, 196  
   przechowywanie ustawień, 201  
   tworzenie wtyczki, 194  
   uaktualnianie metadanych, 195  
   usunięcie metadanych, 196  
 ustawienia widgetu, 84  
 usuwanie  
   błędów, 461  
   komentarzy, 369  
   kod źródłowy, 396  
   metadanych, 322  
   możliwości z roli, 242  
   opcji, 178  
   skryptu, 349  
   użytkowników, 214  
   zadań, 380  
   znaków z adresu URL, 103  
 użytkownik, 210, 230  
   dane użytkownika, 219  
   metadane użytkownika, 223  
   tworzenie, 214  
   uaktualnianie, 214  
   uprawnienia, 232  
   usuwanie, 214  
   wtyczka z metadanymi, 227

## V

Versions, 480

## W

wartość  
   domyślna, 43, 395  
   manage\_music\_collection, 314  
   zwrotna, 205  
 wycięcie, 48  
 wczytywanie tablicy opcji, 178  
 WeblogToolsCollection.com, 503  
 weryfikacja danych, 143, 146, 184  
   ciąg tekstowy adresu e-mail,  
     151  
   dane pochodzące ze  
     zdefiniowanego zbioru, 159  
 HTML, 152  
 JavaScript, 156  
 liczby całkowite, 147  
 mieszane ciągi tekstowe, 148

  oczyszczanie adresów URL, 155  
   oczyszczanie wewnętrznych  
     identyfikatorów, 149  
   pliki cookies, 158  
   tablice danych, 158  
   usuwanie znaczników HTML,  
     153  
   wzorce ciągu tekstowego, 149  
   zapytania do bazy danych, 161  
   zwykłe ciągi tekstowe, 147  
 weryfikacja unikalnego  
   identyfikatora, 139  
 weryfikacji daty, 150  
 węzeł, 154  
   atrybut węzła, 154  
   element węzła, 154  
   tekst węzła, 154  
 węzeł HTML, 154  
 wiadomości RSS, 276  
 widget, 24, 82, 317  
   boj\_widgetexample\_widget\_  
     my\_info(), 84  
   wyświetlający wpisy bloga,  
     441  
   z treścią sieciową, 436  
   zaawansowany, 87  
 widżety kokpitu, 24, 93  
 Williams Brad, 15  
 window.onload, 340  
 witryna  
   stan Archived, 424  
   stan Deleted, 424  
   stan Mature, 424  
   stan Public, 424  
   stan Spam, 424  
 witryna Gravity Forms, 486  
 witryna WordPress.org, 477  
 wizualna integracja, 201  
 własne pola, 319  
 własne typy wpisów bloga, 315  
 właściwości obiektu wpdb, 172  
 WordCamp, 491, 504  
 WordPress, 23  
   wydania główne, 459  
   wydania mniejsze, 459  
 WordPress Development  
   Updates, 502  
 WordPress Meetup, 504  
 WordPress Multisite, *Patrz* sieć  
   Multisite

WordPress Planet, 503  
 WordPress.org, 477  
 wpdb, 172  
 śledzenie liczby zapytań, 172  
 włączenie wyświetlania błędów, 172  
 zmienne klasy, 172  
 WPEngineer.com, 503  
 wstawianie kodu JavaScript, 363  
 wtyczka, 23, 24  
 aktualizacja oprogramowania, 458, 460  
 automatyczny e-mail, 390  
 bezpieczeństwo, 133  
 BOJ Admin Lang, 199  
 buforowanie wpisów, 470  
 dezinstalacja, 44  
 dodawanie użytkownika do wtyczki, 450  
 edycja, 31  
 harmonogram, 383  
 instalacja, 30  
 kanał wiadomości kod źródłowy, 420  
 katalog, 32  
 Kolekcja muzyczna, 334  
 licencja, 39  
 lista sklepów, 407  
 kod źródłowy, 408  
 memcached, 194  
 nagłówek, 38  
 Nieużywane tagi, 140, 142  
 obsługująca własne role i możliwości, 242  
 oczyszczenie danych, 143  
 odnośnik, 363  
 odnośnik bezpośredni, 415  
 oficjalny katalog, 26  
 określanie ścieżek dostępu, 39  
 opłaty, 476  
 pobierająca dane, 287  
 pomoc techniczna, 488  
 poprawione błędy, 465  
 promowanie, 483  
 przełączanie wtyczki, 430  
 punkt końcowy, 418  
 kod źródłowy, 418  
 repozytorium wtyczek, 477  
 Simple Tumblr Backup, 268  
 skrót, 280

testowanie funkcji, 33  
 wtryny w sieci Multisite, 444  
 kod, 445  
 Twitter Info, 264  
 typu wpisu bloga, 334  
 typy, 32  
 unikalne identyfikatory, 136  
 ustawienia, 175, 207  
 usuwanie komentarzy, 369-370, 393  
 kod źródłowy, 396  
 utworzenie pliku, 35  
 weryfikacja, 143  
 wpisy bloga, 435  
 z błędami, 463  
 z metadanymi użytkownika, 227  
 zabezpieczenie, 133  
 zadanie raz w tygodniu, 388  
 zalety, 27  
 zarządzanie, 31  
 zgłoszenie do repozytorium, 479  
 wyrażenia regularne, 149, 270, 403  
 nawiasy kwadratowe, 149  
 znak /, 149  
 znak +, 149  
 wyrażenie `?debug=1`, 135  
 wysyłanie danych do zdalnego API, 267  
 wyszukiwanie funkcji, 495  
 wyszukiwanie zaczepów, 75  
 wyświetlanie komunikatów, 108  
 wyświetlanie widgetu, 439  
 wzorzec, 149  
 wzorzec znacznika, 284

## Y

YSlow, 504

## Z

zaawansowane pole użytkownika, 102  
 zachowanie spójności interfejsu użytkownika, 106  
 zaczep, 51  
 404\_template, 70  
 admin\_menu, 59  
 admin\_notices, 142

akcji admin\_init, 387  
 akcji boj\_cron\_hook, 377  
 akcji init, 329  
 archive\_template, 70  
 attachment\_template, 70  
 author\_template, 70  
 baza danych, 500  
 category\_template, 70  
 comment\_text, 69  
 date\_template, 70  
 filtru, 68, 73  
 filtru map\_meta\_cap, 235  
 front\_page\_template, 70  
 home\_template, 70  
 index\_template, 70  
 init, 59, 136, 408  
 page\_template, 70  
 plugins\_loaded, 58, 136  
 search\_template, 70  
 single\_template, 70  
 tag\_template, 70  
 template\_include, 70  
 template\_redirect(), 60  
 the\_content, 68  
 the\_title, 69  
 uninstall, 45  
 user\_contactmethods, 226  
 wp\_head, 60  
 zaczepy akcji, 58  
 zaczepy zmienne, 75  
 zapisywanie metadanych, 195  
 zapisywanie opcji, 175  
 zapisywanie tablicy opcji, 176  
 zaporą sieciową, 257  
 zapytania, 404  
 zapytania do bazy danych, 161  
 zarządzanie wtyczkami, 31  
 zastąpienie znacznika, 284  
 zbieranie informacji, 489  
 zdalne API, 267  
 zdarzenia w Ajax, 361  
 zdarzenie onload, 341  
 zmiana adresu URL, 401, 407  
 dodanie zmiennej store\_id, 407  
 tworzenie reguły, 407  
 wyczyszczenie reguł, 408  
 zmienna  
 \$boj\_mbe\_image, 102  
 \$clean, 145  
 \$count, 127

\$cron, 382  
 \$date\_format, 382  
 \$deleted, 170  
 \$domain, 445  
 \$local\_posts, 434  
 \$map\_id, 299  
 \$network\_posts, 434  
 \$path, 445  
 \$post\_content, 267  
 \$post\_title, 267  
 \$return\_posts, 433  
 \$schedules, 382  
 \$title, 445  
 \$user\_id, 445  
 dynamiczna, 360  
 globalna \$blog\_id, 427, 450  
 globalna \$current\_user, 220,  
 450  
 znacznik  
 %produkt%, 413  
 <!--more-->, 362

CDATA, 360  
 czasu, 390  
 odnośników bezpośrednich,  
 413  
 zmiany adresów, 413  
 znaczniki nagłówków, 107  
 znaczniki PHP, 49  
 znaczniki typu BBCode, 292  
 znak \$, 338  
 znak ^, 149

## Ż

żądania Ajax, 362, 371  
 żądania FTP, 260  
 żądania HTTP w PHP, 250  
 funkcja fopen(), 250  
 funkcja fsockopen(), 251  
 rozszerzenie CURL, 251  
 rozszerzenie HTTP, 250  
 strumień fopen(), 250

żądania HTTP w WordPress  
 funkcje pomocnicze, 256  
 funkcje wp\_remote\_\*, 252  
 parametry wejściowe, 252  
 wartości zwrotne, 254  
 zakończone niepowodzeniem,  
 254  
 zakończone powodzeniem,  
 255  
 żądania JSONP, 343  
 żądanie  
 do alternatywnego API, 272  
 GET, 252  
 HEAD, 252  
 HTTP, 247  
 odpowiedź na żądanie, 258  
 rejestracja żądań, 258  
 JSON, 343  
 POST, 252, 267

# 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**

Dowiedz się, jak tworzyć własne wtyczki  
— przejdź na wyższy poziom używania systemu WordPress!

**WordPress** to jeden z najpopularniejszych systemów zarządzania treścią. Jest fundamentem niezliczonych blogów, portali i stron WWW. Jego największe atuty to łatwa instalacja, przyjemna konfiguracja i niewygórowane wymagania. Jak to się stało, że zdobył aż taką popularność? Jest coś jeszcze — rozbudowany system wtyczek zwiększających funkcjonalność, dzięki którym możliwości WordPress są w zasadzie nieograniczone!

W trakcie czytania tej książki dowiesz się, jak tworzyć wtyczki i dostosowywać system WordPress do własnych potrzeb, choćby były najbardziej wymyślne. Na początku przygotujesz się do stworzenia własnej wtyczki, a także dowiesz się, jak ją instalować i odinstalowywać oraz dokumentować jej kod. W kolejnych rozdziałach nauczysz się integrować wtyczkę z systemem WordPress — tworzyć widżety, modyfikować menu, obsługiwać pola użytkownika oraz formularze. Jeżeli chcesz, aby Twoja wtyczka zdobyła popularność na całym świecie, koniecznie zapoznaj się ze sposobami tworzenia wtyczek wielojęzycznych. To jednak nie wszystko! Książka porusza wiele innych istotnych kwestii, takich jak bezpieczeństwo wtyczek, przygotowanie strony administracyjnej dla wtyczki, obsługa API. Trzymasz w rękach kompendium wiedzy na temat wtyczek w systemie WordPress. Zaczynaj przygodę z nimi już dziś!

- Szczegóły tworzenia wtyczek o różnym poziomie złożoności — począwszy od bardzo prostych, a skończywszy na wyjątkowo skomplikowanych pluginach typu e-commerce skończywszy.

- Sposoby integracji wtyczek z platformą WordPress, zapisywania ustawień, tworzenia widżetów i skrótów, a także implementacji funkcji deinstalacji wtyczki.

- Stosowanie poprawnych technik przechowywania danych, dostosowywanie ról użytkowników oraz najlepsze praktyki z zakresu implementacji zabezpieczeń we wtyczkach.

- Wykorzystanie procedur pozwalających na używanie własnych typów wpisów na blogu, a także na tworzenie i używanie własnych taksonomii.

- Prezentacja API Http, JavaScript, Ajax, Cron, API Rewrite i wiele innych.

**Brad Williams** to CEO oraz współzałożyciel witryny *webdevstudios.com*, a także autor książki *Professional WordPress* (Wrox, 2010).

**Ozh Richard** to autor wielu popularnych wtyczek dla WordPressa, a także laureat konkursu *Annual WordPress Plugin Competition*.

**Justin Tadlock** opracował wiele wtyczek. Na swojej stronie *ThemeHybrid.com* poświęconej platformie WordPress uczy użytkowników, w jaki sposób używać wtyczek i motywów.

**helion.pl**  
księgarnia internetowa

Nr katalogowy: 7734

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/najczesciej>  
Zamów informacje o nowościach:  
● <http://helion.pl/newsy>

Helion SA  
ul. Kościuszki 1c, 44-100 Gliwice  
tel.: 32 230 98 63  
e-mail: [helion@helion.pl](mailto:helion@helion.pl)  
<http://helion.pl>

**Wrox**  
An Imprint of  
**WILEY**

ISBN 978-83-246-3564-1



Cena: 89,00 zł

sięgnij po WIĘCEJ



KOD KORZYSCI

Informatyka w najlepszym wydaniu