

Spis treści

Przedmowa	19
Wprowadzenie	21
1. Tworzenie aplikacji internetowych w WordPressie	27
Czym jest witryna internetowa?	27
Czym jest aplikacja?	27
Czym jest aplikacja internetowa?	27
Funkcje aplikacji internetowej	28
Aplikacje mobilne	30
Progresywne aplikacje internetowe	30
Dlaczego WordPress?	31
Jesteś już użytkownikiem WordPressa	31
Zarządzanie treścią w WordPressie jest łatwe	31
Łatwe i bezpieczne zarządzanie użytkownikami w WordPressie	32
Wtyczki	32
Elastyczność ma duże znaczenie	33
Częste uaktualnienia zabezpieczeń	33
Koszt	34
Odpowiedź na często pojawiającą się krytykę wybranych aspektów WordPressa	34
Kiedy nie używać WordPressa?	37
Planujesz licencjonować lub sprzedawać technologię witryny internetowej	37
Inna platforma szybciej doprowadzi Cię do celu	38
Elastyczność jest bez znaczenia	38
Aplikacja musi działać w czasie rzeczywistym	39
WordPress jako framework aplikacji	39
WordPress kontra frameworki MVC	40
Anatomia aplikacji internetowej WordPressa	42
Czym jest SchoolPress?	43
SchoolPress działa w sieci zawierającej wiele witryn WordPressa	43

Model biznesowy SchoolPressa	43
Poziomy członkostwa i role użytkowników	44
Klasy są grupami BuddyPress	44
Zadanie to przykład CPT	44
Rozwiązania zadań są podtypami CPT zadań	44
Semestry to taksonomie dla CPT klasy	45
Wydział to taksonomia dla CPT klasy	45
Aplikacja SchoolPress ma jedną główną niestandardową wtyczkę	45
Aplikacja SchoolPress używa kilku innych niestandardowych wtyczek	46
Aplikacja SchoolPress używa motywu Memberlite	46

2. Podstawy WordPressa 47

Struktura katalogu WordPressa	47
Katalog główny	48
/wp-admin	48
/wp-includes	48
/wp-content	48
Struktura bazy danych WordPressa	50
wp_options	50
Funkcje zdefiniowane w /wp-includes/option.php	50
wp_users	53
Funkcje zdefiniowane w plikach /wp-includes/pluggable.php	
i /wp-includes/user.php	53
wp_usermeta	57
wp_posts	61
Funkcje zdefiniowane w /wp-includes/post.php	61
wp_postmeta	66
Funkcje zdefiniowane w /wp-includes/post.php	66
wp_comments	70
Funkcje zdefiniowane w /wp-includes/comment.php	71
wp_commentsmeta	75
Funkcje zdefiniowane w /wp-includes/comment.php	76
wp_terms	78
Funkcje zdefiniowane w /wp-includes/taxonomy.php	78
wp_termmeta	82
wp_term_taxonomy	84
Funkcje zdefiniowane w /wp-includes/taxonomy.php	85
wp_term_relationships	86
Zaczepy — akcje i filtry	87
Akcje	88
Filtry	88

Środowiska programistyczne i hostingowe	90
Praca lokalna	90
Wybór hostingu	91
Środowiska robocze i produkcyjne	92
Rozszerzanie WordPressa	92
3. Stosowanie wtyczek WordPressa	95
Licencja GPLv2	96
Instalowanie wtyczek WordPressa	96
Utworzenie własnej wtyczki	97
Struktura plików we wtyczce	98
/adminpages/	99
/classes/	99
/css/	99
/js/	101
/images/	101
/includes/	102
/includes/lib/	102
/pages/	102
/services/	103
/scheduled/	103
/schoolpress.php	104
Dodatki dla istniejących wtyczek	104
Przypadki użycia i przykłady	104
Pętla WordPressa	105
Zmienne globalne WordPressa	105
Wtyczki bezpłatne	115
Admin Columns	115
Advanced Custom Fields	115
BadgeOS	116
Posts 2 Posts	116
Members	117
W3 Total Cache	117
Yoast SEO	117
Wtyczki premium	118
Gravity Forms	118
BackupBuddy	118
WP All Import	118
Wtyczki społecznościowe	119
BuddyPress	119

4. Motywy	131
Motyw kontra wtyczka	131
Gdzie umieścić kod podczas tworzenia aplikacji?	131
Kiedy opracować wtyczkę?	132
Gdzie umieszczać kod podczas tworzenia motywu?	133
Hierarchia szablonu	133
Szablony strony	135
Przykładowy szablon strony	135
Stosowanie zaczepów do kopiowania szablonów	137
Kiedy należy używać szablonu motywu?	138
Funkcje WordPressa powiązane z motywem	139
Stosowanie funkcji <code>locate_template()</code> w motywach	140
Plik <code>style.css</code>	141
Wersjonowanie plików CSS motywu	142
Plik <code>functions.php</code>	143
Motywy i niestandardowe typy postów	144
Popularne frameworki motywów	144
Frameworki motywów WordPressa	144
Frameworki motywów przeznaczone nie tylko dla WordPressa	146
Tworzenie motywu potomnego dla Memberlite	146
Wykorzystanie frameworka Bootstrap w motywie aplikacji	147
Menu	148
Menu nawigacyjne	148
Menu dynamiczne	149
Responsywny układ strony	150
Wykrywanie urządzenia i ekranu za pomocą CSS	150
Wykrywanie urządzeń i funkcji za pomocą kodu JavaScript	152
Wykrywanie urządzenia w PHP	154
Słowo końcowe na temat wykrywania przeglądarki WWW	157
5. Niestandardowe typy postów, metadane postów i taksonomie	159
Domyślne i niestandardowe typy postów	159
Strona	159
Post	159
Załącznik	159
Wersja	160
Element menu nawigacyjnego	160
Niestandardowe style CSS	160
Changeset	160
Bufor oEmbed	160

Żądania użytkowników	167
Bloki kodu wielokrotnego użycia	167
Definiowanie i rejestrowanie niestandardowych typów postów	167
register_post_type(\$post_type, \$args);	162
Co to jest taksonomia i jak należy z niej korzystać?	173
Taksonomie kontra metadane posta	173
Tworzenie niestandardowych taksonomii	173
register_taxonomy(\$taxonomy, \$object_type, \$args)	173
register_taxonomy_for_object_type(\$taxonomy, \$object_type)	177
Stosowanie niestandardowych typów postów i taksonomii	
we własnych motywach i wtyczkach	177
Szablony stron archiwum i pojedynczego posta w motywie	177
Stare dobre komponenty WP_Query i get_posts()	178
Metadane w niestandardowych typach postów	181
add_meta_box(\$id, \$title, \$callback, \$screen, \$context, \$priority, \$callback_args)	182
Stosowanie elementów obsługi metadanych w edytorze bloków	184
Opakowania klas dla niestandardowych typów postów	185
Rozszerzanie klasy WP_Post kontra opakowanie obiektu tej klasy	187
Po co używać klasy opakowania?	188
CTP i taksonomie będą w jednym miejscu	188
Definiowanie kodu w klasie opakowania	189
Klasa opakowania jest czytelniejsza	191
6. Użytkownicy, role i uprawnienia	193
Pobieranie danych użytkownika	194
Dodawanie, uaktualnianie i usuwanie użytkowników	196
Zaczepy i filtry	199
Czym są role i uprawnienia?	200
Sprawdzanie ról i uprawnień użytkownika	200
Tworzenie niestandardowych ról i uprawnień	202
Rozszerzanie klasy WP_User	203
Dodanie właściwości rejestracji i profilu	205
Dostosowanie do własnych potrzeb tabeli użytkowników w panelu głównym	209
Wtyczki	211
Theme My Login	211
Ukrycie paska administracyjnego przed użytkownikami	
niebędącymi administratorami	212
Paid Memberships Pro	212
Paid Memberships Pro Register Helper	212
Members	213
WP User Fields	213

7. Praca z API WordPressa, obiektami i funkcjami pomocniczymi	215
API skrótów	215
Atrybuty skrótu	216
Skróty zagnieżdżone	217
Usunięcie skrótu	217
Inne użyteczne funkcje powiązane ze skrótami	218
API widżetów	219
Zanim zaczniesz dodawać własny widżet	220
Dodawanie widżetu	220
Definiowanie obszaru widżetu	223
Osadzanie widżetu poza dynamicznym paskiem bocznym	225
API widżetów w panelu głównym WordPressa	226
Usunięcie widżetu panelu głównego	227
Dodawanie własnego widżetu panelu głównego	228
API ustawień	231
Czy naprawdę potrzebna jest strona ustawień?	231
Czy zamiast ustawień można użyć zaczepu lub filtru?	232
Stosowanie standardów podczas dodawania ustawień	233
Ignorowanie standardów podczas dodawania ustawień	233
API przepisywania adresów URL	234
Dodawanie reguły przepisywania adresu URL	235
Usuwanie reguły przepisywania adresu URL	236
Inne funkcje przepisywania adresów URL	237
WP-Cron	239
Definiowanie niestandardowego odstępu czasu	241
Tworzenie harmonogramu dla pojedynczych zdarzeń	241
Wywoływanie zadań mechanizmu cron z serwera	242
Stosowanie zadań mechanizmu cron jedynie po stronie serwera	243
WP Mail	244
Wysyłanie ładniejszych wiadomości e-mail za pomocą WordPressa	245
API nagłówka pliku	246
Dodawanie nagłówków plików do własnych plików	248
Dodawanie nowych nagłówków do wtyczek i motywów	249
API Heartbeat	250
8. Bezpieczny WordPress	255
Dlaczego bezpieczeństwo jest ważne?	255
Podstawy zapewnienia bezpieczeństwa	256
Regularnie uaktualniaj oprogramowanie	256
Nie używaj nazwy użytkownika admin	256
Używaj silnych haseł	256

Przykłady beznadziejnych haseł	257
Przykłady dobrych haseł	257
Zabezpieczenie WordPressa	258
Nie zezwalaj administratorowi na edycję wtyczek lub motywów	258
Zmień domyślny prefiks tabel bazy danych	258
Przenieś wp-config.php	259
Ukryj komunikaty błędów logowania	259
Ukryj numer wersji WordPressa	260
Uniemożliw logowanie poprzez stronę wp-login.php	260
Dodaj niestandardowe reguły .htaccess w celu zabezpieczenia strony wp-admin	261
Certyfikaty SSL i HTTPS	262
Instalacja certyfikatu SSL w serwerze	262
Stosowanie szyfrowania SSL na stronach logowania i administracyjnych	265
Debugowanie problemów związanych z HTTPS	266
Zapobieganie błędom dzięki „opcji nuklearnej”	266
Twórz kopię zapasową całości!	268
Skanuj, skanuj i skanuj!	269
Użyteczne wtyczki zapewnienia bezpieczeństwa	269
Wtyczki związane z blokowaniem spamu	269
Wtyczki związane z tworzeniem kopii zapasowej	270
Wtyczki związane z zaporą sieciową i skanowaniem	270
Wtyczki związane z logowaniem i hasłami	271
Tworzenie bezpiecznego kodu	271
Sprawdzenie uprawnień użytkownika	272
Niestandardowe zapytania SQL	273
Weryfikacja danych, ich oczyszczanie i stosowanie znaków sterujących	273
Jednokrotnie używana liczba	278

9. Frameworki JavaScript285

Co to jest ECMAScript	286
Co to jest ES6	286
Co to jest ES9	287
Co to jest ESNext	287
Co to jest AJAX	287
Co to jest JSON	287
jQuery i WordPress	287
Dodawanie innych bibliotek JavaScript	288
Gdzie umieszczać niestandardowy kod JavaScript	289
Wywołania AJAX za pomocą WordPressa i jQuery	290
Zarządzanie wieloma żadaniami AJAX	295
API Heartbeat	296

Ograniczenia WordPressa związane z przetwarzaniem asynchronicznym	301
Frameworki JavaScript	302
Backbone.js	302
React	303
10. API REST WordPressa	305
Czym jest API REST?	305
API	305
REST	306
JSON	306
HTTP	306
Dlaczego warto używać API REST WordPressa	309
Używanie wersji drugiej API REST WordPressa	311
Odkrycie	311
Uwierzytelnianie	311
Trasy i punkty końcowe	316
Żądania	317
Odpowiedź	320
Dodawanie własnych tras i punktów końcowych	321
register_rest_route(\$namespace, \$route, \$args, \$override);	321
Konfiguracja wtyczki Single Sign-On w WordPressie	322
Dodanie trasy /wp-sso/v1/check	322
Stosowanie uwierzytelniania prostego w omawianej wtyczce	323
Używanie zdefiniowanego punktu końcowego do sprawdzenia danych uwierzytelniających użytkownika	324
Popularne wtyczki używające API REST WordPressa	325
WooCommerce	325
BuddyPress	327
Paid Memberships Pro	328
11. Projekt Gutenberg, bloki i niestandardowe typy postów	333
Edytor WordPressa	334
Wtyczka Classic Editor	335
Używanie bloków podczas tworzenia treści i projektu	335
Używanie bloków do tworzenia funkcjonalności	335
Tworzenie własnego bloku	335
Przykład minimalnego bloku	336
Używanie bloków niestandardowych do tworzenia aplikacji	337
Włączenie edytora bloków w niestandardowych typach postów	338
Kategorie bloków	338
Bloki Homework	339

Ograniczenie bloków do określonych CPT	339
Ograniczenie CPT do określonych bloków	340
Szablon bloku	341
Zapisywanie danych bloku w metadanych posta	342
Podpowiedzi	343
Włączenie WP_SCRIPT_DEBUG	343
Używanie wywołania filemtime() dla wersji skryptu	344
Więcej podpowiedzi	344
Poznaj dokładnie JavaScript, Node.js i React	344
12. Sieć witryn internetowych WordPressa	347
Dlaczego sieć witryn internetowych	347
Dlaczego nie należy korzystać z sieci witryn	348
Alternatywy dla sieci witryn	349
Wielu autorów lub kategorii w tej samej witrynie WordPressa	349
Niestandardowe typy postów	349
Oddzielne witryny internetowe	349
Używanie usługi konserwacji WordPressa	349
Wielodostępność	350
Przygotowanie sieci witryn	350
Zarządzanie siecią witryn WordPressa	352
Panel główny	353
Witryny internetowe	353
Użytkownicy	353
Motywy	354
Wtyczki	354
Ustawienia	355
Uaktualnienia	356
Struktura bazy danych sieci witryn	356
Tabele o zasięgu sieci	356
Tabele poszczególnych witryn	358
Współdzielone tabele witryny internetowej	359
Mapowanie domeny	360
Wtyczki użyteczne w sieci witryn internetowych	360
Gravity Forms User Registration Add-On	361
Dodatek Member Network Sites dla wtyczki Paid Memberships Pro	361
Multisite Global Media	361
Multisite Plugin Manager	361
Multisite Robots.txt Manager	361
NS Cloner — Site Copier	362
WP Multi Network	362

Podstawowa funkcjonalność sieci witryn WordPressa	362
\$blog_id	362
is_multisite()	363
get_current_blog_id()	363
switch_to_blog(\$new_blog)	363
restore_current_blog()	364
get_blog_details(\$fields = null, \$get_all = true)	364
update_blog_details(\$blog_id, \$details = array())	366
get_blog_status(\$id, \$pref)	366
update_blog_status(\$blog_id, \$pref, \$value)	367
get_blog_option(\$id, \$option, \$default = false)	367
update_blog_option(\$id, \$option, \$value)	367
delete_blog_option(\$id, \$option)	368
get_blog_post(\$blog_id, \$post_id)	368
add_user_to_blog(\$blog_id, \$user_id, \$role)	369
wpmu_delete_user(\$user_id)	369
create_empty_blog(\$domain, \$path, \$weblog_title, \$site_id = 1)	370
Funkcje niewymienione w tym podrozdziale	370

13. Lokalizacja aplikacji WordPressa 371

Czy w ogóle zachodzi potrzeba lokalizacji aplikacji	371
Jak lokalizacja jest przeprowadzana w WordPressie	372
Definiowanie lokalizacji w WordPressie	372
Domeny tekstu	373
Definiowanie domeny tekstu	373
Przygotowanie ciągów tekstowych za pomocą funkcji tłumaczeń	375
__(\$text, \$domain = "default")	375
_e(\$text, \$domain = "default")	376
_x(\$text, \$context, \$domain = "default")	376
_ex(\$title, \$context, \$domain = "default")	377
Jednoczesne tłumaczenie tekstu i stosowanie znaków sterujących	377
Tworzenie i wczytywanie plików tłumaczeń	377
Struktura pliku do lokalizacji	378
Generowanie pliku .pot	379
Utworzenie pliku .po	380
Utworzenie pliku .mo	381
GlottPress	381
Używanie narzędzia GlottPress dla wtyczek	
i motywów umieszczanych w repozytorium WordPress.org	381
Utworzenie własnego serwera GlottPress	381

14. Optymalizacja i skalowanie WordPressa	383
Terminologia	383
Źródło kontra krawędź	385
Testowanie	385
Co będzie testowane	386
Pasek debugowania w Chrome	388
Narzędzie Stan witryny WordPressa	390
Apache Bench	390
Siege	397
W3 Total Cache	397
Ustawienia Page Cache	398
Minimalizacja	400
Buforowanie bazy danych	401
Buforowanie obiektów	401
Sieć CDN	402
Kompresja GZIP	402
Hosting	402
Hosting przygotowany z myślą o WordPressie	403
Utworzenie własnego serwera	403
Buforowanie selektywne	416
API Transient	416
Elementy tymczasowe dla wielu witryn internetowych	419
Używanie JavaScriptu do poprawy wydajności działania	420
Tabele niestandardowe	421
Pominięcie WordPressa	423
15. E-commerce	425
Wybór wtyczki	425
WooCommerce	426
Paid Memberships Pro	428
Easy Digital Downloads	429
Bramki płatności	432
Konto sprzedawcy	432
Konfigurowanie modelu SaaS przy użyciu wtyczki Paid Memberships Pro	434
Model SaaS	434
Etap 0. — ustalenie sposobu pobierania opłaty za korzystanie z aplikacji	434
Etap 1. — instalowanie i aktywowanie wtyczki Paid Memberships Pro	435
Etap 2. — ustalenie poziomu członkostwa	435
Etap 3. — konfiguracja stron	437
Etap 4. — wybór ustawień płatności	437
Etap 5. — wybór ustawień wiadomości e-mail	439

Etap 6. — wybór ustawień zaawansowanych	440
Etap 7. — uniemożliwianie dostępu do stron	441
Etap 8. — dostosowanie wtyczki Paid Memberships Pro do własnych potrzeb	443
16. Aplikacje mobilne na bazie WordPressa	449
Przypadki użycia aplikacji mobilnych	449
Natywne i hybrydowe aplikacje mobilne	450
Co to jest natywna aplikacja mobilna	450
Co to jest hybrydowa aplikacja mobilna	451
Dlaczego lepiej wybrać aplikację hybrydową zamiast natywnej	451
Cordova	452
Framework Ionic	457
Opakowanie aplikacji	458
AppPresser	459
17. Biblioteki PHP, integracje usług sieciowych, migracje platform	475
Biblioteki PHP	475
Generowanie i przetwarzanie obrazów	476
Generowanie dokumentu PDF	478
Geolokalizacja i geotargetowanie	483
Kompresja i archiwizowanie plików	485
Narzędzia programistyczne	489
Zewnętrzne API i usługi sieciowe	491
Elasticsearch	491
ElasticPress firmy 10up	491
Google Vision	492
Mapy Google	492
Tłumacz Google	493
Twilio	493
Inne popularne interfejsy API	494
Migracje	495
Migracja hosta	496
Migracja platformy	497
Utworzenie przewodnika mapowania danych	499
18. Przyszłość	501
Jak to było wcześniej	501
API REST	502
Wtyczki WordPressa będą bardziej skoncentrowane na API	502
Headless WordPress	502
GraphQL	503

Projekt Gutenberg	504
Interfejs administracyjny zostanie przeniesiony do rozwiązania opartego na React i Gutenberg	504
Gutenberg zapewni obsługę edycji we frontendzie WordPressa	504
Szablon bloku zastąpi motyw	504
Bloki zastąpią wtyczki	505
Udział WordPressa w rynku będzie się zmieniał	505
WordPress stanie się znacznie popularniejszą platformą do tworzenia aplikacji mobilnych	506
WordPress wciąż będzie użyteczny podczas tworzenia różnych aplikacji internetowych	506