



Zend Framework

od podstaw

Wykorzystaj gotowe rozwiązania PHP
do tworzenia zaawansowanych
aplikacji internetowych

Włodzimierz Gajda



» Idź do

- Spis treści
- Przykładowy rozdział
- Skorowidz

» Katalog książek

- Katalog online
- Zamów drukowany katalog

» Twój koszyk

- Dodaj do koszyka

» Cennik i informacje

- Zamów informacje o nowościach
- Zamów cennik

» Czytelnia

- Fragmenty książek online

» Kontakt

Helion SA
ul. Kościuszki 1c
44-100 Gliwice
tel. 32 230 98 63
e-mail: helion@helion.pl
© Helion 1991–2011

Zend Framework od podstaw. Wykorzystaj gotowe rozwiązania PHP do tworzenia zaawansowanych aplikacji internetowych

Autor: [Włodzimierz Gajda](#)
ISBN: 978-83-246-3052-3
Format: 158×235, stron: 536



Elastyczne PHP – twórz nowe strony z wykorzystaniem Zend Framework!

- Tworzenie stron WWW w Zend Framework – zacznij od „Hello, World”
- Bazy danych i formularze – opanuj wszystkie elementy
- Zabezpieczanie dostępu do aplikacji – nie daj się zaskoczyć

Zend Framework to nowoczesna biblioteka ułatwiająca tworzenie stron WWW w języku PHP. Ten wygodny system pozwala projektantowi stron internetowych znacznie ograniczyć bądź nawet wyeliminować konieczność żmudnego wpisywania kodu na rzecz posługiwania się gotowymi elementami, niezależnie od tego, czy chce zaimplementować podstawowe mechanizmy aplikacji, czy też wzbogacić ją o konkretne funkcjonalności. Wsparcie ze strony twórców języka PHP, stabilna wersja, gotowa do pomocy społeczność programistów-entuzjastów oraz ogromna elastyczność to główne atuty tego rozwiązania, sprawiające, że jego popularność rośnie w dużym tempie.

Niniejsza publikacja ma za zadanie przybliżyć Ci Zend Framework, począwszy od absolutnych podstaw, aż po kwestie związane z pieczołowitym zabezpieczeniem dostępu do aplikacji. Znajdziesz tu jasne i czytelne przykłady zastosowania frameworka w różnych sytuacjach oraz propozycje gotowych rozwiązań konkretnych problemów programistycznych. Nauczysz się tworzyć zarówno proste strony WWW, jak i zaawansowane aplikacje, wymieniać szablony oraz implementować wtyczki. Poznasz sposoby tworzenia i wykorzystywania klas, współpracy z bazą danych, publikowania aplikacji w Internecie oraz używania formularzy. Krótko mówiąc, masz w rękę kompletny przewodnik po jednym z najlepszych frameworków PHP!

- Pierwszy projekt w Zend Framework i praca w środowisku NetBeans
- Tworzenie i usuwanie kontrolerów oraz akcji
- Wymiana szablonu HTML/CSS i dołączanie zewnętrznych zasobów
- Zasoby i implementacja inicjalizującej je wtyczki
- Bazy danych, tabele i relacje
- Identyfikacja rekordów na podstawie wartości slug
- Menu generowane na podstawie zawartości tabeli bazy danych
- Publikowanie aplikacji wykorzystującej bazę danych na serwerze hostingowym
- Przetwarzanie formularza, czyli implementacja interfejsu CRUD i dostosowywanie kodu HTML
- Walidatory oraz filtry i przesyłanie plików na serwer
- Zabezpieczanie haseł funkcjami skrótów
- Rejestracja i ograniczanie uprawnień użytkowników
- Modularyzacja aplikacji

Programuj swobodnie, wykorzystując uniwersalne biblioteki PHP!

Spis treści

Część I	Tworzenie stron WWW w Zend Framework	9
Rozdział 1.	Pierwszy projekt w Zend Framework	11
	Podsumowanie	17
	Uruchomienie gotowego przykładu	18
Rozdział 2.	Praca w środowisku NetBeans	19
Rozdział 3.	Tworzenie i usuwanie kontrolerów oraz akcji	27
	Praca w środowisku deweloperskim	33
Rozdział 4.	Wymiana szablonu HTML/CSS	35
	Adresy strony z wierszem	41
Rozdział 5.	Dołączanie zewnętrznych zasobów	43
	Analiza kodu HTML generowanego przez aplikację	47
Rozdział 6.	Hiperłącza	49
	Zalety mapowania adresów wewnętrznych/zewnętrznych	51
	Reguły domyślne i funkcja pomocnicza url()	60
Rozdział 7.	Strona błędu 404	63
	Analiza odpowiedzi HTTP	71
Rozdział 8.	Publikowanie projektu na serwerze hostingowym	75
	Zestawienie plików tworzących projekt 8.1	80
Rozdział 9.	Podsumowanie	89
	Pliki źródłowe aplikacji	90
	Przebieg wykonania aplikacji	90
	Konwencje nazewnictwa klas i plików	95
Część II	Procedura inicjalizacji aplikacji	97
Rozdział 10.	Zasoby	99
	Zasoby i ich opcje konfiguracyjne	102
	Zasób db	102
	Zasób frontController	103
	Zasób layout	103
	Zasób router	104
	Zasób view	105

Kiedy automatyczna konfiguracja zawodzi?	107
Metody <code>_init()</code> klasy <code>Bootstrap</code>	108
Inicjalizacja wybranych zasobów	109
Dostęp do zainicjalizowanych zasobów	110
Dostęp do zainicjalizowanych zasobów wewnątrz akcji	111
Kiedy nie implementować własnej metody <code>_init()</code> ?	111
Rozdział 11. Implementacja wtyczki inicjalizującej zasoby	123
Klasa wtyczki i klasa zasobu	123
Włączanie przetwarzania wtyczki	126
Rozdział 12. Podsumowanie	133
Część III Zend_DB — klasy zapewniające dostęp do baz danych ...	135
Rozdział 13. Pierwszy projekt ZF wykorzystujący bazę danych	137
Uruchomienie gotowego projektu	154
Rozdział 14. Klasa <code>Zend_Db_Adapter_Abstract</code> i klasy pochodne	157
Klasa <code>Zend_Db</code>	158
Klasa <code>Zend_Db_Adapter_Abstract</code> i jej klasy pochodne	160
Metoda <code>Zend_Db_Adapter_Abstract::fetchRow()</code>	163
Metoda <code>Zend_Db_Adapter_Abstract::fetchAll()</code>	165
Metoda <code>Zend_Db_Adapter_Abstract::fetchCol()</code>	166
Metoda <code>Zend_Db_Adapter_Abstract::fetchOne()</code>	167
Metoda <code>Zend_Db_Adapter_Abstract::fetchAssoc()</code>	167
Metoda <code>Zend_Db_Adapter_Abstract::fetchPairs()</code>	167
Metoda <code>Zend_Db_Adapter_Abstract::setFetchMode()</code>	168
Metoda <code>Zend_Db_Adapter_Abstract::insert()</code>	169
Metoda <code>Zend_Db_Adapter_Abstract::lastInsertId()</code>	169
Metoda <code>Zend_Db_Adapter_Abstract::delete()</code>	170
Metoda <code>Zend_Db_Adapter_Abstract::update()</code>	171
Metoda <code>Zend_Db_Adapter_Abstract::query()</code>	172
Metoda <code>Zend_Db_Adapter_Abstract::quote()</code>	172
Metoda <code>Zend_Db_Adapter_Abstract::quoteInto()</code>	173
Metody do obsługi transakcji	173
Użycie wyrażeń SQL	174
Rozdział 15. Klasa <code>Zend_Db_Table</code> i klasy z nią związane	181
Klasa <code>Zend_Db_Select</code>	181
Klasa <code>Zend_Db_Table</code>	185
Konstruktor klasy <code>Zend_Db_Table</code>	186
Metoda <code>Zend_Db_Table::insert()</code>	187
Metoda <code>Zend_Db_Table::delete()</code>	188
Metoda <code>Zend_Db_Table::update()</code>	189
Metoda <code>Zend_Db_Table::find()</code>	189
Metoda <code>Zend_Db_Table::select()</code>	190
Metoda <code>Zend_Db_Table::fetchAll()</code>	190
Metoda <code>Zend_Db_Table::fetchRow()</code>	191
Metoda <code>Zend_Db_Table::createRow()</code>	191
Klasa <code>Zend_Db_Table_Row</code>	193
Klasa <code>Zend_Db_Table_Rowset</code>	194

Rozdział 16. Dostosowywanie klas dostępu do bazy danych	201
Modyfikacja porządku kolekcji rekordów zwracanych przez metodę Zend_Db_Table::fetchAll()	202
Definiowanie własnej metody __toString() w klasach dziedziczących po Zend_Db_Table_Row	203
Rozdział 17. Relacje 1:n (jeden do wielu)	211
Klucze główne	211
Relacja jeden do wielu	211
Relacje 1:n w programie MySQL Workbench	213
Klucze obce o wartości NULL	215
Akcje referencyjne	216
Użycie relacji 1:n w Zend Framework	217
Operowanie rekordami powiązanych relacją	219
Tworzenie rekordów	219
Rekordy zależne	220
Rekord nadrzędny	222
Implementacja własnych metod dostępu do rekordów powiązanych	223
Rozdział 18. Relacje n:m (wiele do wielu)	229
Relacja wiele do wielu	229
Relacje n:m w programie MySQL Workbench	230
Użycie relacji n:m w Zend Framework	230
Operowanie rekordami powiązanych relacją	233
Tworzenie rekordów	233
Rekordy zależne	234
Implementacja własnych metod dostępu do rekordów powiązanych	235
Rozdział 19. Podsumowanie	243
Część IV Szczegółowe dane rekordu	247
Rozdział 20. Akcja show — wyświetlanie szczegółowych danych rekordu	249
Metoda identyfikacji i wyszukiwania rekordów w bazie danych	249
Akcja show i jej adres	250
Przetwarzanie w akcji show	250
Generowanie adresów stron akcji show	251
Konfiguracja przyjaznych adresów akcji show	258
Rozdział 21. Identyfikacja rekordów na podstawie wartości slug	263
Klasa konwertująca polskie znaki	264
Funkcje string2slug() oraz html2slug()	266
Automatyczne generowanie wartości slug podczas zapisywania rekordu w bazie danych ..	284
Rozdział 22. Widoki częściowe	289
Rozdział 23. Menu generowane na podstawie zawartości tabeli bazy danych	303
Rozdział 24. Zapisywanie w bazie danych plików binarnych	309
Nagłówek Content-Type	309
Konwersja rozszerzenia w typ mime	310
Wyłączenie przetwarzania widoków .phtml	311
Modyfikacja odpowiedzi w akcji	312

Rozdział 25. Publikowanie aplikacji wykorzystującej bazę danych na serwerze hostingowym	321
Rozdział 26. Podsumowanie	339
Część V Formularze	341
Rozdział 27. Formularz i kontrolki	343
Tworzenie formularzy poleceniem zf create form	343
Tworzenie kontrolek formularza	344
Umieszczanie formularza na stronach WWW	346
Rodzaje kontrolek	348
Klasa Zend_Form_Element_Button	350
Klasa Zend_Form_Element_Captcha	350
Klasa Zend_Form_Element_Checkbox	351
Klasa Zend_Form_Element_File	351
Klasa Zend_Form_Element_Hash	352
Klasa Zend_Form_Element_Hidden	352
Klasa Zend_Form_Element_Image	352
Klasa Zend_Form_Element_Multi	353
Klasa Zend_Form_Element_MultiCheckbox	353
Klasa Zend_Form_Element_Multiselect	354
Klasa Zend_Form_Element_Password	354
Klasa Zend_Form_Element_Radio	354
Klasa Zend_Form_Element_Reset	355
Klasa Zend_Form_Element_Select	355
Klasa Zend_Form_Element_Text	356
Klasa Zend_Form_Element_Textarea	356
Klasa Zend_Form_Element_Submit	356
Rozdział 28. Przetwarzanie formularza, czyli implementacja interfejsu CRUD	359
Formularz Application_Form_Imie	360
Operacje dwuetapowe	361
Tworzenie nowego rekordu	361
Edycja rekordu	362
Akcje interfejsu CRUD	362
Adresy URL akcji CRUD	363
Akcja index	363
Akcja createform	365
Akcja create	366
Akcja delete	368
Akcja edit	369
Akcja update	370
Akcja show	372
Parametryzacja kontrolera CRUD	375
Implementacja klasy My_Crud_Controller	378
Rozdział 29. Dostosowywanie kodu HTML formularzy	383
Domyślny kod HTML formularza klasy Zend_Form	383
Funkcje pomocnicze formularzy	384
Obiekty dekorujące i przebieg dekoracji	386
Domyślne dekoratory klasy Zend_Form	388
Domyślne dekoratory klasy Zend_Form_Element	389
Proces generowania kodu HTML formularza	390

Modyfikacja elementów form oraz dl	391
Modyfikacja kodu HTML kontrolek formularza	392
Modyfikacja identyfikatora znacznika form	395
Definiowanie szablonu formularza	396
Rozdział 30. Walidatory i filtry	399
Filtrowanie i walidacja kontrolek formularza	400
Interfejs Zend_Filter_Interface	400
Klasa Zend_Validate_Abstract i jej pochodne	402
Filtry i walidatory kontrolek	404
Tytuł książki	404
Rok wydania	405
Badanie numeru miesiąca	406
Badanie zależności pomiędzy kilkoma kontrolkami	407
Sprawdzanie liczb rzymskich od I do X	409
Walidacja kontrolek checkbox	410
Sprawdzanie numeru ISBN	411
Sprawdzanie poprawności liczby typu float	411
Walidacja przy użyciu wyrażeń regularnych	411
Walidator kontrolki CAPTCHA	412
Rozdział 31. Przesyłanie plików na serwer	417
Rozdział 32. Edycja zależności relacyjnych	427
Klucze obce NOT NULL	427
Klucze obce NULL	437
Rozdział 33. Podsumowanie	445
Część VI Zabezpieczanie dostępu do aplikacji	447
Rozdział 34. Pierwsza aplikacja zawierająca formularz do logowania	449
Umieszczanie kont użytkowników w bazie danych	450
Dodawanie konta	450
Formularz do logowania	452
Kontroler autoryzacyjny	454
Czy użytkownik jest zalogowany?	457
Zabezpieczanie dostępu do danych tylko dla zalogowanych użytkowników	457
Rozdział 35. Zabezpieczanie haseł funkcjami skrótu	461
Funkcja md5()	461
Funkcja sha1()	462
Skróty haseł	462
Zmodyfikowana tabela user	463
Dodawanie konta	464
Kontroler autoryzacyjny	465
Rozdział 36. Wysyłanie poczty	471
Wysyłanie poczty przy użyciu Zend_Mail	471
Obiektowa implementacja wysyłania listów z hasłami	474
Rozdział 37. Rejestracja użytkowników	477
Rejestracja użytkownika w systemie	478
Resetowanie zapomnianego hasła	482
Zmiana hasła	487

Rozdział 38. Ograniczanie uprawnień użytkowników	493
Przykładowe uprawnienia	494
Czy użytkownik ma uprawnienia do wykonania akcji?	494
Implementacja klasy My_Crud_Auth_Controller	496
Modyfikacja kontrolera AuthController	498
Polecenia nadające uprawnienia	499
Ustalanie uprawnień poleceniami	502
Polecenia allow-action-access i disallow-action-access	502
Polecenia grant i revoke	502
Polecenia set-readable i set-unreadable	502
Polecenia grant-editor-rules i revoke-editor-rules	502
Polecenia grant-reader-rules i revoke-reader-rules	503
Polecenie clear	503
Rozdział 39. Modularyzacja aplikacji	507
Nazewnictwo klas zawartych w modułach	508
Adresy URL akcji w module	509
Rozdział 40. Podsumowanie	517
Część VII Dodatki	519
Dodatek A Użycie Doctrine w aplikacji Zend Framework	521
Instalacja zf-doctrine	521
Skorowidz	527

Rozdział 4.

Wymiana szablonu HTML/CSS

Teraz zajmiemy się udekorowaniem aplikacji gotowym szablonem HTML/CSS. Najpierw podzielimy widoki na dwa fragmenty: układ oraz treść. Wyodrębniony układ będzie zapisany w jednym pliku *layout.phtml*, widoki akcji będą zaś generowały wyłącznie treść zawartą na stronie. W ten sposób uprościmy nadanie spójnego wyglądu wszystkim stronom aplikacji.

Dekorację aplikacji pisanej w ZF szablonem HTML/CSS rozpoczynamy od wydania polecenia:

```
zf enable layout
```

W wyniku wydania powyższej komendy w folderze aplikacji pojawi się plik:

```
application/layouts/scripts/layout.phtml
```

zaś w pliku *configs/application.ini* pojawi się wpis:

```
resources.layout.layoutPath = APPLICATION_PATH "/layouts/scripts/"
```

Powyższa reguła modyfikuje przebieg wykonania aplikacji: generowanie strony WWW wysyłanej do przeglądarki będzie odbywało się przy użyciu pliku *layout.phtml*¹.

W pliku *layout.phtml* umieszczamy szablon strony WWW, czyli m.in. znaczniki `html`, `head` i `body` oraz elementy ustalające podział na nagłówek, menu, treść czy panele (np. `div`). Treść zawarta na stronie WWW jest generowana przez widoki akcji, które tym razem nie zawierają już elementów `html`, `head` czy `body`.

¹ Jeśli instrukcję zawartą w pliku *application.ini* zakomentujesz, poprzedzając ją średnikiem:

```
;resources.layout.layoutPath = APPLICATION_PATH "/layouts/scripts/"
```

wówczas plik *layout.phtml* nie będzie wykorzystywany. Generowanie stron WWW będzie przebiegało identycznie jak w pierwszych trzech rozdziałach. Kod HTML wysyłany do przeglądarki będzie w całości pochodził z widoku akcji.

Przeanalizujmy stronę WWW, która była generowana w przykładzie trzecim jako wynik przetwarzania akcji `wierszyk/pokaz` (listing 3.3). W widoku akcji zawarty był kompletny kod HTML strony wysyłanej do przeglądarki.

Wprowadzając podział na szablon i treść, plik z listingu 3.3 dzielimy na dwa oddzielne pliki: szablon `layout.phtml` widoczny na listingu 4.1 oraz widok przedstawiony na listingu 4.2.

Listing 4.1. Szablon `layout.phtml` dla strony z listingu 3.3

```
<!DOCTYPE...>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pl" lang="pl">
  <head>
    <title>Wylicznarka</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  </head>
  <body>
    <?php echo $this->layout()->content ?>
  </body>
</html>
```

Listing 4.2. Treść wyodrębniona ze strony 4.1

```
<h1>Ene, due</h1>
<p>
Ene, due, rike, fake<br />
Torbe, borbe, ósme, smake<br />
Eus, deus, kosmateus<br />
I morele baks.
</p>
```

Zwróć uwagę, że szablon aplikacji widoczny na listingu 4.1 zawiera specjalną instrukcję:

```
<?php echo $this->layout()->content ?>
```

Powoduje ona umieszczenie w szablonie wyniku przetwarzania akcji. W ten sposób podczas wykonywania aplikacji dwa oddzielne pliki z listingów 4.1 oraz 4.2 są łączone w jeden dokument z listingu 3.3, który jest następnie wysyłany do przeglądarki.

Przykład 4. Wierszyk pt. Dwa kabele

Przygotuj stronę WWW, która będzie prezentowała treść wiersza pt. *Dwa kabele*. Zadanie wykonaj tak, by wierszyk był prezentowany na stronie akcji `index/index`. W rozwiązaniu utwórz szablon `layout.phtml`. Układ strony WWW zapisz w pliku `layout.phtml`, treść wiersza zaś — w widoku akcji `index/index` (tj. w pliku `index.phtml`).

Wykorzystaj szablon zawarty w folderze `html-css-template/`. Treść wiersza oraz folder `html-css-template/` są zawarte w pliku `04-start.zip`, który znajdziesz pod adresem <ftp://ftp.helion.pl/przyklady/ZendFr.zip>. Wykonana strona powinna wyglądać tak jak na rysunku 4.1.



Rysunek 4.1. Strona z wierszem pt. *Dwa kabele*

ROZWIĄZANIE

Krok 1. Utwórz nowy projekt ZF

Utwórz nowy projekt PHP, który wykorzystuje ZF. Projekt nazwij `dwa-kabele`. Utworzenie nowego projektu będzie równoważne wykonaniu komendy:

```
zf create project . dwa-kabele
```

Następnie w pliku `public/.htaccess` dodaj reguły:

```
SetEnv APPLICATION_ENV development
DirectoryIndex index.php
```



Wskazówka

W zależności od swoich preferencji wszystkie projekty omówione w książce możesz wykonać w wierszu poleceń lub w środowisku NetBeans.

Krok 2. Włącz przetwarzanie szablonu `layout.phtml`

Wydaj komendę:

```
zf enable layout
```

W folderze aplikacji pojawi się plik *application/layouts/scripts/layout.phtml*. Jego zawartość jest przedstawiona na listingu 4.3.

Listing 4.3. *Domyślna zawartość pliku layout.phtml*

```
<?php echo $this->layout()->content ?>
```

Krok 3. Dostosuj widok akcji index/index

W widoku akcji *index/index* wprowadź treść wierszyka. Zarys pliku *application/views/scripts/index/index.phtml* jest przedstawiony na listingu 4.4.

Listing 4.4. *Fragment widoku index.phtml*

```
<h2>Włodzimierz Gajda</h2>
<h3>Dwa kabele</h3>
<p>
  Czasem tak sie dziwnie składa.<br />
  Że gdy nic nie zapowiada<br />
  Żadnych nieszczęść czy frustracji.<br />
  Jakiś smyk wkroczy do akcji<br />
  I, być może, bez złych chęci.<br />
  Sielankę ojcu zamąci.<br />
<br />
...

```

Krok 4. Zmień szablon HTML/CSS

W pliku *layout.phtml* dodaj znaczniki HTML ustalające wygląd generowanej strony WWW. Zadanie mamy rozwiązać, wykorzystując szablon zapisany w folderze *03-start/html-css-template/*. Plik *04-start/html-css-template/index.html* jest przedstawiony na listingu 4.5.

Listing 4.5. *Szablon index.html, którego chcemy użyć w projekcie dwa-kabele*

```
<!DOCTYPE...>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pl" lang="pl">
  <head>
    <title>template</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <link rel="stylesheet" type="text/css" href="css/style.css" media="screen" />
    <link rel="stylesheet" type="text/css" href="css/print.css" media="print" />
  </head>
  <body>

  <div id="pojemnik">
    <h1 id="naglowek">Wiersze i wierszyki...</h1>
    <div id="tekst">
      <p>
        Lorem ipsum...
      </p>
    </div>
  </div>

```

```
<div id="dol"></div>
</div>

</body>
</html>
```

Znaczniki z listingu 4.3 należy umieścić w pliku *layout.phtml*, tak jak pokazano na listingu 4.6.

Listing 4.6. Szablon *layout.phtml*, który spowoduje, że strona z wierszem będzie wyglądała tak jak na rysunku 4.1

```
<!DOCTYPE...>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="pl" lang="pl">
  <head>
    <title>Wierszyk</title>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <link rel="stylesheet" type="text/css" href="css/style.css" />
    <link rel="stylesheet" type="text/css" href="css/print.css" media="print" />
  </head>
  <body>

  <div id="pojemnik">
    <h1 id="naglowek">Wiersze i wierszyki...</h1>
    <div id="tekst">
      <?php echo $this->layout()->content(); ?>
    </div>
    <div id="dol"></div>
  </div>

  </body>
</html>
```

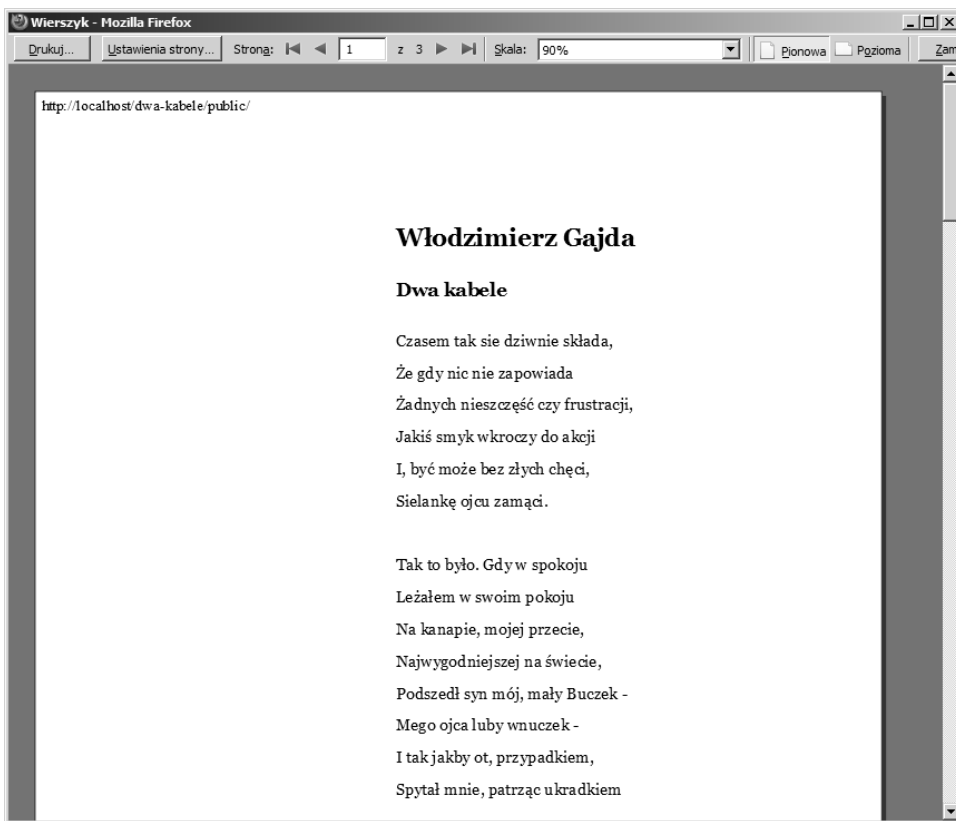
Kod HTML generowanej strony WWW należy sformatować stylami CSS zawartymi w plikach *html-css-template/css/style.css* oraz *html-css-template/css/print.css*. Style z pliku *style.css* są przeznaczone dla monitorów, a style z pliku *print.css* — dla urządzeń drukarek. Pliki *style.css* oraz *print.css* kopiujemy z folderu *html-css-template/css/* do folderu *public/css*². Modyfikowanie szablonu aplikacji kończymy, kopiując pliki graficzne z folderu *html-css-template/images/* do folderu *public/images/*.

W celu sprawdzenia wyglądu aplikacji odwiedź za pomocą przeglądarki adres:

```
http://localhost/dwa-kabele/public/
```

Powinieneś ujrzeć stronę taką jak na rysunku 4.1. Jeśli chcesz poznać funkcję stylów *print.css*, wykonaj w przeglądarce operację *Plik/Podgląd wydruku*. Ujrzysz taką stronę jak na rysunku 4.2.

² Foldery *public/css/* oraz *public/images/* należy utworzyć samodzielnie.



Rysunek 4.2. Podgląd wydruku strony z rysunku 4.1

Dzięki zastosowaniu w pliku *print.css* właściwości:

```
display: none
```

elementy dekoracyjne zostały usunięte i nie pojawiają się na wydruku.



Wskazówka

Pliki graficzne tworzące szablon HTML/CSS z rysunku 4.1 nazywają się *gora.png*, *naglowek.png*, *pojemnik.png* i *stopka.png*. Pliki te nie występują w kodzie HTML, a wyłącznie w kodzie CSS. W projekcie nie korzystamy ze znacznika `img`:

```
<img src="" alt="" />
```

Pliki PNG stanowią tło wybranych elementów HTML i są dołączane do strony przy użyciu właściwości `background`, np.:

```
body {
    background: #e8edef url('../images/gora.png') repeat-x;
    margin: 0;
    font-family: Georgia, serif;
}
```

Ścieżki do plików graficznych są względne. Prowadzą z pliku *style.css* do folderu *images/*.

Adresy strony z wierszem

W wykonanej aplikacji strona akcji `index/index` jest dostępna pod sześcioma adresami URL:

```
http://localhost/dwa-kabele/public/  
http://localhost/dwa-kabele/public/index.php  
http://localhost/dwa-kabele/public/index.php/  
http://localhost/dwa-kabele/public/index.php/index  
http://localhost/dwa-kabele/public/index.php/index/index  
http://localhost/dwa-kabele/public/index.php/index/index/
```

Tylko dwa pierwsze adresy będą powodowały wyświetlenie strony WWW ozdobionej stylami CSS. Cztery dolne adresy, czyli:

```
http://localhost/dwa-kabele/public/index.php/  
http://localhost/dwa-kabele/public/index.php/index  
http://localhost/dwa-kabele/public/index.php/index/index  
http://localhost/dwa-kabele/public/index.php/index/index/
```

będą wyświetlały stronę WWW pozbawioną stylów CSS. Dzieje się tak dlatego, że ścieżka prowadząca do stylów CSS jest zapisana w pliku *layout.phtml* jako względna:

```
<link rel="stylesheet" type="text/css" href="css/style.css" />
```

Style zostaną więc odnalezione wyłącznie wtedy, gdy znajdują się w stosunku bieżącego adresu URL w podfolderze *css/*.

Skorowidz

- #, 57
- \$_belongsTo, 343
- \$_decorators, 387
- \$_element, 387
- \$_name, 148
- \$_page, 129
- \$bind, 163, 171, 172, 173
- \$cfg, 120
- \$defaultsArray, 119
- \$fetchMode, 163
- \$form, 417
- \$router, 110
- \$sql, 163, 172, 244
- \$table, 171
- \$where, 171, 173
- .htaccess, 13, 14, 25, 63, 83
 - __construct(), 185
 - __toString(), 183, 201, 203, 339
 - __initAdresy(), 111
 - __insert(), 339
 - __update(), 339
- 404 (błąd), 63

A

- abbr(), 266
- action helpers, 303
- addDecorator(), 387, 393
- addElement(), 344, 346
- admin, 503
- adres
 - domenowy, 81
 - konwersja, 50
 - wewnętrzny, 49, 51
 - zewnętrzny, 49, 51
- akcja, 17
 - show, 249, 250, 289, 309
- akcje referencyjne, 216
- allow-action-access, 499, 502

- Alnum, 401, 403
- Alpha, 401, 403
- analiza odpowiedzi HTTP, 71
- animacja, 44
- API, 243
- appendBody(), 312
- appendContent(), 452
- application, 12
- application.ini, 90, 99, 101, 105, 107, 113, 123, 133
- APPLICATION_ENV, 92
- Application_Form_Imie, 360
- APPLICATION_PATH, 91
- atak typu
 - brute-force, 468
 - SQL Injection, 165
- Auth, 457
- AuthController, 454, 465
- AUTO INCREMENT, 142
- automatyczne generowanie wartości slug, 339

B

- BaseName, 401
- basePath, 105
- baseUrl(), 46, 48, 56, 67
- baza danych (projekt), 139
- beginTransaction(), 162
- Between, 403
- biblioteki Dojo, 101
- bind(), 182
- blokowanie dostępu do plików, 84
- błąd 404, 63
- błędny adres URL, 63
- Boolean, 401
- Bootstrap, 108, 109
- bootstrap(), 109
- Bootstrap.php, 12, 94, 95, 133
- brute-force, 468

C

cachemanager, 101
 CamelCaseToUnderscore, 402
 CASCADE, 216
 charset, 105
 checkbox, 410
 clear, 499, 503
 clearDecorators(), 387
 clearElements(), 346
 collate, 146
 columns(), 182
 commit(), 163
 Compress, 401
 configs, 12
 contentType, 105
 Controller, 13
 controllers, 12
 count(), 194
 create, 361, 366
 CREATE
 SCHEMA IF NOT EXISTS, 146
 TABLE, 146
 createform, 361, 365
 createRow(), 191
 Cross Site
 Request Forgery, 399
 Scripting, 399
 CRUD, 359, 427, 445
 current(), 194

D

DaneProvider, 318
 Database/Forward Engineer, 144, 175
 Date, 403
 db, 101, 102
 DbTable, 148
 Deccription, 389
 Decompress, 401
 Decrypt, 401
 default Action, 33
 defaultAttachOrder, 117
 defaultControllerName, 33
 define(), 91
 delete, 368
 delete(), 162, 170, 173, 186, 188, 193
 dependent table, 212
 Description, 391
 destination table, 212
 Digits, 401, 403
 Dir, 401
 DirectoryIndex, 25

dirname(), 92
 disallow-action-access, 499, 502
 display: none, 40
 distinct(), 182
 długie nazwy metod, 223
 docs, 13
 Doctrine (biblioteka), 521
 doctype, 105
 Dojo, 101
 domena, 81, 86
 domyślna
 akcja, 59
 strona, 15, 16
 dostęp do
 akcji (zabezpieczenie), 493
 aplikacji, 517
 bazy danych, 101, 176, 243
 plików, 83
 DROP SCHEMA IF EXISTS, 146

E

edit, 362, 369
 edycja rekordu, 362
 elementy dekoracyjne, 40
 EmailAddress, 403
 encje (konwersja), 266
 Encrypt, 401
 error, 12, 13, 63
 Errors, 389, 391

F

F6, 22
 factory(), 158
 fetchAll(), 149, 161, 165, 173, 178, 186, 190, 201, 339
 fetchAssoc(), 161, 167
 fetchCol(), 161, 166, 173
 fetchOne(), 161, 167, 173
 fetchPairs(), 162, 167, 173
 fetchRow(), 161, 164, 173, 191, 249
 filtrowanie, 399
 find(), 186, 189, 249
 findOneBySlug(), 339
 Float, 403
 fluent interface, 184
 flush privileges, 146
 foreign key, 211
 form, 383
 Form, 391, 395
 formatowanie stylami CSS, 39
 formButton(), 384

formCheckbox(), 384
 FormElements, 391
 formTextarea(), 384
 formularz
 do logowania, 452
 edycyjny, 427
 na stronie WWW, 346
 rejestracyjny, 478
 forUpdate(), 182
 from(), 182
 frontController, 101
 frontController (opcje konfiguracyjne), 103
 funkcja
 pomocnicza, 303
 skrótu, 461, 517

G

Generate DROP SCHEMA, 144
 get(), 125
 getAdapter(), 185
 getConnection(), 159
 getDecorator(), 387
 getDecorators(), 387
 getElement(), 346
 getElements(), 346
 getInvokeArg(), 111
 getMimeTypeOnExt, 310
 getPage(), 129
 getResource(), 110, 111
 getResponse(), 312
 getRow(), 194
 getTableOfContents(), 273
 getValues(), 361, 400
 Gmail, 472
 grant, 499, 502
 grant all, 146
 grant-editor-rules, 499, 502
 grant-reader-rule, 499
 grant-reader-rules, 503
 GreaterThan, 403
 group(), 182

H

hash table, 229
 haslo konta root, 147
 headLink(), 106, 108
 headMeta(), 107
 headScript(), 106, 108
 headStyle(), 106
 headTitle(), 106, 108, 117
 heaving(), 182

Hello, world!, 11
 helpers, 12
 Hex, 403
 hiperłącze, 57
 Hostname, 403
 href, 57
 html2slug(), 266
 HtmlEntities, 401
 htmlGetFirstMatch(), 275
 HtmlTag, 389, 391
 HTTP/1.1 404 Not Found, 72

I

identyfikacja aplikacji, 101
 implementacja własnej wtyczki, 123, 133
 importowanie bazy danych, 333
 InArray, 403
 index, 12, 13, 363, 454
 index.php, 13, 14, 95
 index.phtml, 12
 indexAction(), 13, 177
 IndexController.php, 95
 inicjalizacja zasobów, 133
 init(), 125, 129
 inlineScript(), 106
 insert(), 149, 162, 169, 173, 186, 187
 insertIfNotExists(), 339
 Int, 401, 403
 interfejs
 API, 243
 do rejestracji użytkowników, 477
 dostępu do baz danych, 243
 obiektowy, 245
 tablicowy, 244, 245
 Ip, 403
 Isbn, 403
 iso2utf8(), 264
 ISO-8859-2, 264
 isPost(), 361
 isValid(), 361, 400

J

join(), 183
 joinCross(), 183
 joinFull(), 183
 joinFullUsing(), 183
 joinInner(), 183
 joinLeft(), 183
 joinLeftUsing(), 183
 joinNatural(), 183
 joinRight(), 183

joinRightUsing(), 183
 joinUsing(), 183
 jQuery LightBox, 417

K

klasa, 13
 klucz
 główny, 142, 211, 360
 obcy, 211, 215, 427, 437
 kod
 biblioteki Dojo, 101
 SQL zaprojektowanej bazy danych, 144
 kodowanie, 140
 kolekcja kontrolek, 343
 komunikat
 diagnostyczny, 452
 o błędach, 33
 konfiguracja
 modułów aplikacji, 101
 poczty elektronicznej, 101
 sesji, 101
 zasobów aplikacji (ręczna), 108
 konto, 450
 kontroler, 17
 aplikacji, 13
 usuwanie, 30
 kontrolki formularza, 344
 konwencje nazewnicze, 95
 konwersja
 adresu, 50, 51
 obiektu w napis, 339
 korzystanie w jednej aplikacji z kilku połączeń z
 bazami danych, 101

L

Label, 389, 391
 lastInsertId(), 162, 169
 layout, 99, 101, 103, 107
 layout.phtml, 101, 114, 121
 LessThan, 403
 library, 13
 limit(), 183
 limitPage(), 183
 litery duże na małe, 266
 LiveHTTPHeaders, 71, 310
 loadDefaultDecorators(), 387
 locale, 101
 log, 101
 login, 454, 455
 logout, 454

M

mail, 101
 mailNewPassword(), 474
 mapa witryny, 101
 mapowanie adresów URL, 66
 md5(), 461, 462
 mechanizm przywracania dostępu, 471
 menu, 303
 główne, 339
 nawigacyjne, 101
 merge(), 120
 metoda uruchamiania projektu, 21
 mime, 310
 mod_rewrite, 14
 models, 12
 modules, 101
 moduł, 507
 adresy akcji, 509
 nazwa klasy, 508
 multodb, 101, 243
 My_Crud_Auth_Controller, 496
 My_Crud_Controller, 378, 496
 My_Mail, 474
 My_Mail_Gmail, 474
 My_Mail_Netart, 474
 My_Mime, 310
 My_Page, 126
 My_Validate_User, 485
 MySQL Workbench, 139, 213, 230

N

nagłówek HTTP, 71
 należy do, 343
 named parameters, 165
 navigation, 101
 nazwa
 akcji, 14
 bazy danych (dywiz), 175
 domyślnego kontrolera, 33
 domyślnej akcji, 33
 konta, 480
 metody, 14
 pliku .php, 14
 projektu, 20
 widoku, 14
 NetBeans, 19
 NO ACTION, 216
 notacja wielbłądzich garbów, 28
 NotEmpty, 403
 nowe konto, 464
 Null, 401
 NULL, 215

O

obiekt nadrzędny, 343
 obsługa
 błędnych żądań HTTP, 15
 błędu 404, 65
 transakcji, 173
 odpowiedzi wysyłane przez serwer WWW, 71
 odwołania, 43
 błędne, 60
 odzyskiwanie hasła, 517
 opcje konfiguracyjne, 102
 operacja
 C, 361
 U, 362
 order(), 183
 orHeaving(), 183
 orWhere(), 183
 owner, 343

P

page, 126
 page.extension = "php", 121
 page.title.content, 117
 page.title.separator, 117
 pamięć podręczna, 101
 parent table, 212
 parseArticle(), 274
 partial view, 289
 partial(), 289
 pdo_mysql, 158
 placeholders, 165
 poczta elektroniczna, 471
 pojemniki (w kodzie SQL), 165
 pokazAction(), 31
 polskie znaki diakrytyczne, 266
 połączenie
 w trybie leniwym, 159
 z kilkoma bazami, 243
 z serwerem bazodanowym (instrukcje), 243
 populate(), 361
 porządek rekordów (modyfikacja), 339
 positional parameters, 165
 PostCode, 403
 powiązania relacyjne, 339
 n:m, 232
 preDispatch(), 457, 517
 PregReplace, 401
 primary key, 211
 projekt (utworzenie nowego), 11
 przetwarzanie szablonu, 37
 layout.phtml, 89

przyjazne adresy, 339
 public, 13, 25
 function 173
 publikacja witryny w Internecie, 75
 Put NetBeans metadata into a separate directory, 23

Q

query(), 162, 172
 quote(), 162, 172
 quoteInto(), 162, 173

R

randomPassword(), 482
 reader, 503
 RealPath, 401
 realpath(), 92
 referential actions, 216
 Regex, 411
 RegEx, 403
 reguła, 58
 konfiguracyjna, 49
 konfigurująca wirtualny serwer stron WWW, 81
 translacji, 61, 77, 258
 rekord nadrzędny, 222
 rekordy
 powiązane relacją, 219
 zależne, 220, 234, 339
 relacje, 245
 1:n, 211, 213, 215
 n:m, 229, 230
 removeDecorator(), 387
 render(), 386, 387
 reset(), 183
 resetowanie hasła, 471, 482
 resources, 99
 RESTRICT, 216
 revoke, 499, 502
 revoke-editor-rules, 499, 502
 revoke-reader-rules, 499, 503
 ręczna konfiguracja zasobów aplikacji, 108
 robot internetowy, 101
 rollBack(), 163
 root, 147, 503
 router, 95, 99, 101, 104, 108
 routing, 51
 rozszerzenia, 309
 Run Project, 22

S

salt, 463
 save(), 193
 scripts, 12
 seek(), 194
 select(), 163, 186, 190
 send(), 474
 session, 101
 SET
 DEFAULT, 217
 NULL, 216
 setAction(), 361
 setDecorators(), 387
 setDefaultAdapter(), 185
 setDefaultAttachOrder(), 106
 setElements(), 346
 setFetchMode(), 162, 168
 setFromArray(), 193
 setHeader(), 312
 setMethod(), 361
 set-readable, 499, 502
 setSeparator(), 106
 setSlug(), 286, 339
 set-unreadable, 499, 502
 sha1(), 461, 462
 show, 249, 250, 289, 309, 372
 showAction(), 339
 simplexml_load_file(), 151
 skrócenie generowanego napisu do zadanej
 długości, 266
 skrypt wypełniający bazę danych, 207
 slug, 284, 339
 source table, 212
 spis treści, 276
 sposób uruchamiania projektu, 21
 SQL, 174
 injection, 399
 string2slug(), 266
 StringLength, 403
 StringToLower, 401
 StringToUpper, 401
 StringTrim, 401
 StripNewLines, 401
 StripTags, 401
 strona błędu 404, 69
 SyntaxHighlighter, 270
 system uprawnień, 493, 517
 szablon
 HTML/CSS, 35
 layout.phtml, 114
 strony WWW, 35

Ś

ścieżki dostępu, 85
 środowisko NetBeans, 19

T

tabela
 docelowa, 212
 haszująca relacji, 229
 skrzyżowań, 229
 źródłowa relacji, 212
 tablica asocjacyjna, 153
 napisów, 244
 tests, 13
 tłumaczenie na inne języki, 101
 toArray(), 193, 194
 translacja adresów, 57, 101, 298
 translate, 101
 tryb leniwy, 272
 try-catch, 159
 tworzenie
 akcji, 89
 kontrolera, 89
 nowego
 konta, 464
 projektu, 20, 89
 rekordu, 361
 typ
 danych, 309
 mime, 310
 tytuł strony, 78

U

udostępnianie zasobów (na stronie WWW), 309
 unauthorizedAction(), 498
 UnderscoreToCamelCase, 402
 unikalny tytuł strony, 79
 union(), 183
 update, 362, 370
 update(), 162, 171, 173, 186, 189
 uprawnienia, 493,
 do wykonania akcji, 494
 uproszczone napisy, 339
 useragent, 101
 userHasAccess(), 494
 Usuń, 30
 usuwanie
 akcji, 89
 kontrolera, 30, 89
 utf8, 146

UTF-8, 264
 utf8_polish_ci, 140, 146
 utworzenie nowego projektu, 11

V

view, 101, 105, 123
 helper, 303
 ViewHelper, 389, 391
 views, 12

W

walidacja, 399
 where(), 183
 widok
 akcji, 38
 aplikacji, 101
 częściowy, 289, 339
 wielojęzyczne wersje aplikacji, 101
 WINDOWS-1250, 264
 własne
 implementacje metod, 223
 metody, 111, 133
 wtyczki, 123
 właściciel, 343
 wstawianie rekordu do
 bazy danych, 339
 tabeli, 149
 wypełnianie bazy danych rekordami, 152
 wyrażenia SQL, 174
 wysyłanie poczty elektronicznej, 471
 wyszukiwanie rekordów
 na podstawie wartości slug, 339
 o zadanej wartości klucza głównego, 249
 wyświetlenie komunikatu: Brak dostępu, 498

Z

zabezpieczanie dostępu do
 aplikacji, 449
 danych, 457
 zapisywanie informacji diagnostycznych, 101
 zasoby zewnętrzne, 43
 zawartość tabeli bazy danych, 149
 zdjęcie, 44
 Zend_Form_Element_File, 417
 Zend_Application, 94, 95
 Zend_Application_Bootstrap_Bootstrap, 95
 Zend_Application_Resource_View, 123
 Zend_Controller_Action, 95
 Zend_Controller_Response_Http, 312
 Zend_Db, 157, 158, 243, 521

Zend_Db::FETCH_ASSOC, 168
 Zend_Db::FETCH_BOTH, 168
 Zend_Db::FETCH_NUM, 168
 Zend_Db::FETCH_OBJ, 168
 Zend_Db_Adapter_Abstract, 157, 160, 244
 Zend_Db_Adapter_Db2, 158
 Zend_Db_Adapter_Mysqli, 158
 Zend_Db_Adapter_Oracle, 158
 Zend_Db_Adapter_Pdo_Ibm, 157
 Zend_Db_Adapter_Pdo_Mssql, 157
 Zend_Db_Adapter_Pdo_Mysql, 157
 Zend_Db_Adapter_Pdo_Oci, 157
 Zend_Db_Adapter_Pdo_Pgsql, 157
 Zend_Db_Adapter_Pdo_Sqlite, 157
 Zend_Db_Adapter_Sqlsrv, 158
 Zend_Db_Expr, 157
 Zend_DB_Expr, 174
 Zend_Db_Select, 157, 181, 182
 Zend_Db_Statement_Exception, 187
 Zend_Db_Table, 157, 185, 186, 243
 Zend_Db_Table_Relationships, 157
 Zend_Db_Table_Row, 157, 193, 243
 Zend_Db_Table_Rowset, 157, 194, 243
 Zend_Filter_Interface, 400
 Zend_Form_Element, 348, 400, 445
 Zend_Form_Element_Text, 348
 Zend_Form_Element_Button, 350
 Zend_Form_Element_Captcha, 350
 Zend_Form_Element_Checkbox, 351
 Zend_Form_Element_File, 351
 Zend_Form_Element_Hash, 352
 Zend_Form_Element_Hidden, 352
 Zend_Form_Element_Image, 352
 Zend_Form_Element_MultiCheckbox, 353
 Zend_Form_Element_Multiselect, 354
 Zend_Form_Element_Password, 354
 Zend_Form_Element_Radio, 354
 Zend_Form_Element_Reset, 355
 Zend_Form_Element_Select, 355
 Zend_Form_Element_Submit, 348, 356
 Zend_Form_Element_Text, 356
 Zend_Form_Element_Textarea, 356
 Zend_Mail, 471
 Zend_Validate_Abstract, 402
 Zend_Validate_Db_NoRecordExists, 403
 Zend_Validate_File_Exists, 403
 Zend_Validate_File_NotExists, 403
 zf create action, 89
 zf create controller, 89
 zf create form, 445
 zf create project, 12, 89
 zf enable layout, 56, 89, 114
 zf-doctrine, 521
 zfproject.xml, 13

złośliwy kod, 399

zmiana hasła, 487

znaczniki HTML (usuwanie), 266

znaki różne od liter i cyfr (zastępowanie
separatorem), 266

Programuj swobodnie, wykorzystując uniwersalne biblioteki PHP!

Zend Framework to nowoczesna biblioteka ułatwiająca tworzenie stron WWW w języku PHP. Ten wygodny system pozwala projektantowi stron internetowych znacznie ograniczyć bądź nawet wyeliminować konieczność żmudnego wpisywania kodu na rzecz posługiwania się gotowymi elementami, niezależnie od tego, czy chce zaimplementować podstawowe mechanizmy aplikacji, czy też wzbogacić ją o konkretne funkcjonalności. Wsparcie ze strony twórców języka PHP, stabilna wersja, gotowa do pomocy społeczność programistów-entuzjastów oraz ogromna elastyczność to główne atuty tego rozwiązania, sprawiające, że jego popularność rośnie w dużym tempie.

Niniejsza publikacja ma za zadanie przybliżyć Ci Zend Framework, począwszy od absolutnych podstaw, skończywszy na kwestiach związanych z pieczołowitym zabezpieczaniem dostępu do aplikacji. Znajdziesz tu jasne i czytelne przykłady zastosowania frameworka w różnych sytuacjach oraz propozycje gotowych rozwiązań konkretnych problemów programistycznych. Nauczysz się tworzyć zarówno proste strony WWW, jak i zaawansowane aplikacje, wymieniać szablony oraz implementować wtyczki. Poznasz sposoby tworzenia i wykorzystywania klas, współpracy z bazą danych, publikowania aplikacji w internecie oraz używania formularzy. Krótko mówiąc, masz w ręku kompletny przewodnik po jednym z najlepszych frameworków PHP!

- Pierwszy projekt w Zend Framework i praca w środowisku NetBeans
- Tworzenie i usuwanie kontrolerów oraz akcji
- Wymiana szablonu HTML/CSS i dołączanie zewnętrznych zasobów
- Zasoby i implementacja inicjalizującej je wtyczki
- Bazy danych, tabele i relacje
- Identyfikacja rekordów na podstawie wartości sług
- Menu generowane na podstawie zawartości tabeli bazy danych
- Publikowanie aplikacji wykorzystującej bazę danych na serwerze hostingowym
- Przetwarzanie formularza, czyli implementacja interfejsu CRUD i dostosowywanie kodu HTML
- Walidatory oraz filtry i przesyłanie plików na serwer
- Zabezpieczanie haseł funkcjami skrótów
- Rejestracja i ograniczanie uprawnień użytkowników
- Modularyzacja aplikacji

nr katalogowy: 6821

Księgarnia internetowa:
<http://helion.pl>

Zamówienia telefoniczne:
0 801 339900
0 601 339900

helion.pl
księgarnia internetowa

Sprawdź najnowsze promocje:
• <http://helion.pl/promocje>
Książki najchętniej czytane:
• <http://helion.pl/najbestsellery>
Zamów informacje o nowościach:
• <http://helion.pl/newsy>



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



Cena 89,00 zł

ISBN 978-83-246-3052-3



9 788324 630523

Informatyka w najlepszym wydaniu