

Łukasz Sosna

# yiiframework

---

Wykorzystaj Yii,  
a w mgnieniu oka zbudujesz  
wydajną stronę WWW!

Jak działa Yii — od czego zacząć i na czym skończyć budowę aplikacji?  
Jakie możliwości oferuje Yii i co zyska dzięki nim programista?  
Do czego służy Yii — jaką aplikację stworzysz za jego pomocą?



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.

Redaktor prowadzący: Ewelina Burska  
Projekt okładki: Studio Gravite/Olsztyn  
Obarek, Pokoński, Pazdrijowski, Zaprucki

Materiały graficzne na okładce zostały wykorzystane za zgodą Shutterstock.

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)

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

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

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Materiały do książki można znaleźć pod adresem:

<ftp://ftp.helion.pl/przyklady/yiifra.zip>

ISBN: 978-83-246-7920-1

Copyright © Helion 2014

Printed in Poland.

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

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

# Spis treści

<b>Wstęp</b> .....	<b>7</b>
<b>Rozdział 1. Pobieranie</b> .....	<b>9</b>
Wymagania frameworku .....	10
<b>Rozdział 2. Instalacja</b> .....	<b>13</b>
Dodawanie ścieżki dostępu w systemie Windows .....	13
Dodawanie ścieżki dostępu w systemie Linux .....	14
Uruchomienie frameworku .....	14
Plik .htaccess .....	16
„Tyle pracy, aby wygenerować jedną stronę!” .....	17
<b>Rozdział 3. Model, widok, kontroler</b> .....	<b>19</b>
Wzorzec model – widok – kontroler .....	19
Kontroler .....	19
Parametry akcji .....	21
Model .....	22
Baza danych .....	23
DAO .....	24
Przekazywanie danych .....	26
Parametry w zapytaniu .....	29
Połączenie z wieloma bazami danych .....	30
ActiveRecord .....	32
Porównanie ActiveRecord, SQL i Query Builder .....	43
Widok .....	43
<b>Rozdział 4. Tworzenie formularzy</b> .....	<b>47</b>
Budowa formularza .....	47
Model .....	48
Widok .....	49
Kontroler .....	50
Elementy formularza .....	52
Pole tekstowe — text .....	52
Pole z ukrytą treścią — password .....	54
Wielowierszowe pole tekstowe — textarea .....	55
Pole listy rozwijanej — select .....	56
Pole opcji (przełącznik) — radio .....	58
Pole wyboru — checkbox .....	59

Selektor plików — file .....	61
Pole ukryte — hidden .....	63
Przycisk — button .....	64
Przycisk wysłania formularza — submit .....	65
Przycisk przywracania danych domyślnych formularza — reset .....	67
Walidacja danych .....	68
Puste pola .....	70
Porównywanie .....	71
Poprawność adresu e-mail .....	73
Format daty i czasu .....	73
Domyślna wartość pola .....	74
Wartości istniejące w tabeli .....	75
Konwertowanie otrzymanych danych .....	76
Dozwolone ciągi znakowe .....	77
Dozwolona liczba wpisanych znaków .....	78
Dozwolone wartości liczb .....	79
Sprawdzanie typu zmiennej .....	80
Wartość unikalna .....	81
Walidacja adresu URL .....	81
Sprawdzanie wartości pola w zależności od scenariusza .....	82
<b>Rozdział 5. Zaawansowana konfiguracja .....</b>	<b>87</b>
Zmiana adresu URL .....	87
Zmiana tytułu .....	87
Ładowanie bibliotek i komponentów .....	88
Błędy aplikacji .....	90
Dzienniki systemowe .....	90
Tworzenie łączy .....	94
Usuwanie nazwy pliku index.php z adresu URL .....	98
Zmiana ścieżki dostępu Yii .....	99
Własne dane konfiguracyjne .....	100
Strony statyczne .....	101
Przekierowania .....	103
Dodawanie komunikatów w sesji .....	103
Korzystanie z szablonów .....	105
Przygotowanie aplikacji do opublikowania .....	108
<b>Rozdział 6. Gii .....</b>	<b>109</b>
Uruchomienie Gii .....	109
Połączenie z bazą danych .....	110
Tabela w bazie danych .....	110
Logowanie do Gii .....	111
Tworzenie modeli .....	112
Tworzenie kontrolerów .....	114
Tworzenie formularzy .....	116
Tworzenie systemu CRUD .....	118
<b>Rozdział 7. Zaawansowane możliwości Yii .....</b>	<b>125</b>
Automatyczne dzielenie na strony .....	125
Sortowanie danych .....	128
CAPTCHA w formularzach .....	130
Zapisywanie plików na serwerze poprzez formularz .....	134

<b>Rozdział 8. Zii .....</b>	<b>137</b>
Rozszerzenia Zii .....	137
CMenu .....	137
CListView .....	138
CDetailView .....	140
CBreadcrumbs .....	141
CGridView .....	143
<b>Rozdział 9. Budujemy blog — interfejs AR .....</b>	<b>145</b>
Baza danych .....	145
Tworzenie aplikacji .....	152
Główny plik aplikacji .....	153
Główny plik konfiguracyjny .....	153
Plik .htaccess .....	154
Arkusze stylów .....	155
Główny plik szablonu strony .....	155
Model dla tabeli blog_kategorie .....	156
Model dla tabeli blog_komentarze .....	158
Model dla tabeli blog_uzytkownicy .....	159
Model dla tabeli blog_wpisy .....	161
Model do obsługi zmiany hasła .....	162
Kontroler bloga .....	163
Kontroler logowania .....	167
Kontroler panelu administratora .....	170
Widok logowania .....	179
Widok bloga — strona główna .....	181
Widok bloga — kategoria .....	182
Widok bloga — wpis .....	184
Widok panelu — kategorie .....	186
Widok panelu — aktualizacja kategorii .....	189
Widok panelu — komentarze .....	190
Widok panelu — wpisy .....	191
Widok panelu — aktualizacja wpisu .....	193
Widok panelu — zmiana hasła .....	196
Słowo końcowe .....	198
<b>Rozdział 10. Budujemy blog — zapytania SQL .....</b>	<b>199</b>
Baza danych .....	199
Utworzenie aplikacji .....	199
Główny plik aplikacji .....	200
Główny plik konfiguracyjny .....	200
Plik .htaccess .....	201
Arkusze stylów .....	201
Główny plik szablonu strony .....	202
Model dla tabeli blog_kategorie .....	203
Model dla tabeli blog_komentarze .....	205
Model dla tabeli blog_uzytkownicy .....	208
Model dla tabeli blog_wpisy .....	209
Model do obsługi zmiany hasła .....	212
Kontroler bloga .....	214
Kontroler logowania .....	218
Kontroler panelu administratora .....	220
Widok bloga — strona główna .....	229
Widok bloga — kategoria .....	231

---

Widok bloga — wpis .....	231
Widok logowania .....	234
Widok panelu — kategorie .....	236
Widok panelu — aktualizacja kategorii .....	238
Widok panelu — komentarze .....	240
Widok panelu — wpisy .....	241
Widok panelu — aktualizacja wpisu .....	245
Widok panelu — zmiana hasła .....	247
Słowo końcowe .....	249
<b>Rozdział 11. Debugowanie aplikacji .....</b>	<b>251</b>
Co pracuje szybciej: AR czy SQL? .....	254
Opcje debugera .....	255
<b>Podsumowanie .....</b>	<b>257</b>
<b>Skorowidz .....</b>	<b>259</b>

# Rozdział 8.

## Zii

### Rozszerzenia Zii

Biblioteka rozszerzeń Zii jest dostarczana wraz z systemem Yii. Te dodatkowe rozszerzenia, napisane przez zespół twórców frameworku, pozwalają generować w prosty sposób dodatkowe elementy. Jest to spore ułatwienie w tworzeniu stron WWW.

### CMenu

Klasa CMenu umożliwia efektywne generowanie menu aplikacji. Można za jej pomocą stworzyć menu zdefiniowane w postaci listy, jak również własne szablony prezentacji elementów nawigacyjnych.

Aby sprawdzić, jak to działa, utworzymy kontroler *MenuController.php* w katalogu *protected/controllers*. Zadeklarujemy akcję *actionIndex()*, której zadaniem jest wczytanie widoku (listing 8.1).

**Listing 8.1.** Kontroler wczytujący plik widoku

---

```
<?php

class MenuController extends Controller
{
    public function actionIndex()
    {
        $this->render('index');
    }
}

?>
```

---

Elementy Zii zastosujemy w widoku, co umożliwi oddzielenie w aplikacji warstwy prezentacji od warstwy treści. Zaraz opiszę, jak to konkretnie wygląda. Najpierw utworzymy plik widoku *index.php* w katalogu *protected/views/menu*. Za pomocą metody `widget()` ładujemy rozszerzenie — klasę `CMenu`. Pierwszym parametrem metody `widget()` jest ścieżka dostępu do włączanej wtyczki (`CMenu`), natomiast w drugim parametrze definiujemy właściwości tworzonego menu w tablicy. Każde menu zaczynamy od klucza `items`, do którego przypisujemy tablicę z dwoma wymaganymi elementami. Pierwszym elementem jest klucz `label` wraz z opisem (etykietą) łącza, a drugim — klucz `url` z adresem URL do strony, którą ma uruchomić kliknięcie danej pozycji menu. Musimy tutaj pamiętać zarówno o kontrolerze, jak i o wywoływanej metodzie, w przeciwnym razie dojdzie do błędów. Jeśli chcemy uzyskać menu z podmenu, jako kolejny parametr dodajemy `items`, w którym definiujemy poszczególne elementy podmenu, dokładnie tak samo jak napisaliśmy menu główne (listing 8.2).

**Listing 8.2.** Widok do generowania menu. Zwróć uwagę na załadowaną wtyczkę `CMenu`

```
<?php

$this->widget('zii.widgets.CMenu', array(
    'items'=>array(
        array('label'=>'Strona główna', 'url'=>array('stronaglowna/index')),
        array('label'=>'Oferta', 'url'=>array('oferta/index'),
            'items'=>array(
                array('label'=>'Produkt 1', 'url'=>array('produkty/produkt1')),
                array('label'=>'Produkt 2', 'url'=>array('produkty/produkt1')),
                array('label'=>'Produkt 3', 'url'=>array('produkty/produkt1')),
            )
        ),
        array('label'=>'Zaloguj', 'url'=>array('zaloguj/index')),
    )
));

?>
```

Jeśli chcesz obejrzeć, jak wygląda nowo utworzona strona, w przeglądarce internetowej wpisz adres <http://localhost/yii/test/menu>.

## CListView

Klasa `CListView` umożliwia proste prezentowanie danych pobranych na przykład z bazy danych. Umożliwia wykorzystywanie szablonów widoków do wyświetlania danych, co pozwala na łatwe dostosowywanie wyglądu generowanej strony do potrzeb. Co istotne, `CListView` zapewnia obsługę zarówno sortowania, jak i podziału zestawu wyników na strony.

Najpierw napiszemy kontroler. Utworzymy plik *ListaController.php*, który zapiszemy w katalogu *protected/controllers*. Dane, które będziemy prezentować użytkownikowi, już powinieneś mieć zapisane w tabeli nazwiska. Jeśli tak nie jest, w rozdziale 6. znajdziesz informacje o strukturze tabeli i zawartych w niej danych. Jeśli natomiast tabela



nazwiska istnieje, dzięki odwołaniu do klasy `CActiveDataProvider()`, gdzie w pierwszym parametrze podajemy nazwę modelu przeznaczonego do obsługi tabeli, używamy możliwość wykorzystania z niej danych. Następnie ładujemy widok i przekazujemy do niego odwołanie do bazy danych (listing 8.3).

---

**Listing 8.3.** Kontroler wczytujący dane pochodzące z modelu

---

```
<?php

class ListaController extends Controller
{
    public function actionIndex()
    {
        $Dane = new CActiveDataProvider('Nazwiska');

        $this->render('index', array(
            (
                'Dane' => $Dane,
            )
        ));
    }
}

?>
```

---

Następnie utworzymy widok. Znajdzie się on w pliku *index.php* w katalogu *protected/views/lista*. Najpierw musimy załadować wtyczkę *CListView*. W pierwszym parametrze podajemy ścieżkę dostępu do wtyczki, a w drugim definiujemy wymagane parametry: element *dataProvider* wraz z uchwytym do danych, element *itemView* z szablonem prezentacji treści oraz *sortableAttributes*, do którego przypiszemy tablicę zawierającą nazwy kolumn, według których odbędzie się sortowanie danych (listing 8.4).

---

**Listing 8.4.** Widok z użyciem wtyczki *CListView*

---

```
<?php

$this->widget('zii.widgets.CListView', array(
    'dataProvider'=>$Dane,
    'itemView'=>'_zobacz',
    'sortableAttributes'=>array(
        'id',
        'nazwisko',
    ),
));

?>
```

---

Utwórzmy jeszcze widok do prezentacji jednej linii treści. Znajdzie się on w pliku *\_zobacz.php* w folderze *protected/view/lista*. W widoku należy zdefiniować bloki (sekcje), w których zostaną wyświetlone dane. Dane te zostaną tu przekazane dzięki zastosowaniu obiektu *\$data* z odwołaniem do pola z tabeli (listing 8.5).

**Listing 8.5.** Szablon przygotowany do wyświetlenia jednego rekordu z bazy danych

```
<?php

echo '<div class="view">';

echo '<div>'.$data->id.'</div>';

echo '<div>'.$data->nazwisko.'</div>';

echo '</div>';

?>
```

Strona już jest gotowa. Możesz ją uruchomić, wpisując w pasku adresu: <http://localhost/yii/test/lista>.

## CDetailView

Klasa `CDetailView` umożliwia ustawienie sposobu wyświetlania określonych szczegółów pojedynczego rekordu pobranego z bazy danych.

Pracę rozpoczniemy od zdefiniowania kontrolera. Utwórzmy więc plik *SzczegolyController.php* w katalogu *protected/controllers*. Wybierzmy jeden rekord z modelu *Nazwiska*; w tym celu możesz się posłużyć jego kluczem głównym. Następnie przekażmy dane do widoku (listing 8.6).

**Listing 8.6.** Kontroler przekazujący dane do pliku widoku

```
<?php

class SzczegolyController extends Controller
{
    public function actionIndex()
    {
        $Model = Nazwiska::model()->findByPk(1);

        $this->render('index', array
        (
            'Model' => $Model,
        )
        );
    }
}

?>
```

Teraz trzeba jeszcze tylko utworzyć plik widoku, dzięki któremu będzie można wyświetlić dane. Utworzymy plik *index.php* w katalogu *protected/views/szczegoly*. W pliku wczytamy wtyczkę *CDetailView* oraz zadeklarujemy jej właściwości. Pierwszym parametrem jest *data*; będzie on przechowywał dane otrzymane z bazy danych. Kolejnym jest tablica *attributes()* zawierająca pola do wyświetlenia (listing 8.7).

**Listing 8.7.** Widok wyświetlający szczegółowe dane na temat rekordu.

```
<?php

$this->widget('zii.widgets.CDetailView', array(
    'data'=>$Model,
    'attributes'=>array(
        'id',
        'ileosob',
        'nazwisko',
    ),
));
?>
```

Klasa *CDetailView* umożliwia również wykorzystanie mechanizmu relacji w bazie danych: w wygenerowanym kodzie mogą się znaleźć wartości pobrane z innych tabel, zgodnie ze zdefiniowanymi relacjami. Więcej informacji na ten temat znajdziesz na stronie internetowej aplikacji w sekcji *Class Reference*<sup>1</sup>.

Jeśli chcesz sprawdzić, jak działa nowo utworzona strona, wpisz w pasku adresu przeglądarki internetowej: *http://localhost/yii/test/szczegoly*.

## CBreadcrumbs

Klasa *CBreadcrumbs* umożliwia wyświetlenie grupy łączy wskazujących pozycję bieżącej strony w całej aplikacji. Jest to przydatny element, zwłaszcza kiedy tworzy się bardzo rozbudowaną witrynę.

Aby wypróbować działanie tej wtyczki, utworzymy kontroler, który posłuży do załadowania pliku widoku. Zapiszemy więc plik *SciezkaController.php* w katalogu *protected/controllers*. W kontrolerze należy wczytać metodę do renderowania widoku wraz z plikiem (listing 8.8).

**Listing 8.8.** Kontroler służący do załadowania widoku

```
<?php

class SciezkaController extends Controller
{
    public function actionIndex()
    {
```

<sup>1</sup> <http://www.yiiframework.com/doc/api/> — przyp. red.

```

        $this->render('index');
    }
}
?>

```

Teraz należy utworzyć widok w pliku *index.php* w katalogu *protected/views/sciezka*. W widoku załadujemy wtyczkę. W pierwszym parametrze określimy nazwę klasy *CBreadcrumbs* wraz ze ścieżką dostępu, natomiast w drugim zadeklarujemy tablicę zawierającą klucz *links*, do którego przypiszemy tablicę zawierającą wyświetlane kolejno nazwy poszczególnych części witryny. Każda taka nazwa stanowi klucz tablicy, w której należy zdefiniować ścieżki dostępu wraz z niezbędnymi parametrami. Pierwszy z tych parametrów to nazwa kontrolera i metody, a kolejnymi są zmienne wraz z wartościami (listing 8.9).

**Listing 8.9.** Widok z definicją łączy wskazujących pozycję bieżącej strony w witrynie

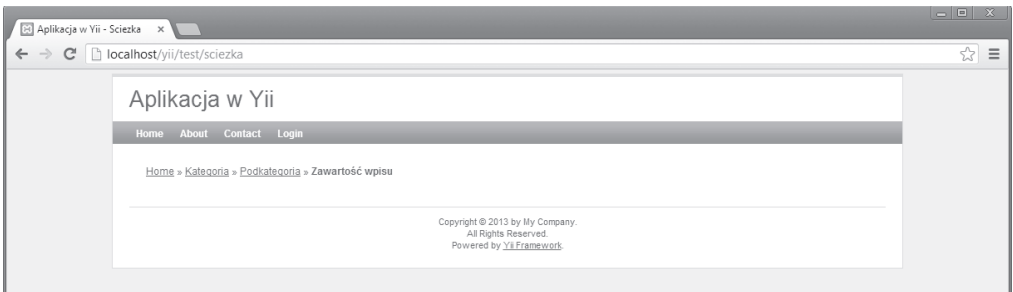
```

<?php

$this->widget('zii.widgets.CBreadcrumbs', array(
    'links'=>array(
        'Kategoria'=>array('sciezka/kategoria', 'id'=>12),
        'Podkategoria'=>array('sciezka/podkategoria', 'id'=>25),
        'Zawartość wpisu',
    ),
));
?>

```

Gotowa strona znajduje się pod adresem *http://localhost/yii/test/sciezka*. Wywołanie strony z przeglądarki spowoduje wygenerowanie łączy wskazujących pozycję przeglądanej strony w aplikacji (rysunek 8.1).



**Rysunek 8.1.** Dzięki wyświetlonej ścieżce użytkownik łatwo się zorientuje, z której części aplikacji właśnie korzysta

## CGridView

Klasa `CGridView` umożliwia wyświetlanie zestawu danych pobranych z bazy danych i sformatowanych w formie tabeli. Każdy wiersz takiej wyświetlonej tabeli odpowiada pojedynczej pozycji danych, a każda kolumna — pojedynczemu atrybutowi danych.

`CGridView` zapewnia zarówno sortowanie danych, jak i podział większego zestawu rekordów na kilka stron. Funkcjonalności te mogą działać i z wykorzystaniem Ajaksa, i w trybie zwykłych żądań strony. Co ciekawe, jeśli użytkownik wyłączy w przeglądarce obsługę JavaScript, sortowanie i podział danych na strony będzie się automatycznie odbywało w trybie zwykłych żądań strony.

Wtyczkę `CGridView` należy stosować wraz z dostawcą danych.

Sprawdźmy, jak to działa. Tworzymy kontroler i zapisujemy go w pliku *SiatkaController.php* w katalogu *protected/controllers*. Musimy dodać dowiązanie do modelu wybierającego dane z tabeli w naszej bazie danych. Następnie ładujemy widok, do którego przekazujemy to dowiązanie (listing 8.10).

---

**Listing 8.10.** *Kontroler przekazujący dane do widoku*

---

```
<?php

class SiatkaController extends Controller
{
    public function actionIndex()
    {
        $Dane = new CActiveDataProvider('Nazwiska');

        $this->render('index', array
            (
                'Dane' => $Dane,
            )
        );
    }
}

?>
```

---

Pozostał nam jeszcze do napisania widok, dzięki któremu będziemy mogli wyświetlić informacje pobrane z bazy. Tworzymy plik widoku *index.php* w katalogu *protected/views/siatka*. W widoku ładujemy wtyczkę za pomocą metody `widget()`. W pierwszym parametrze tej metody podajemy nazwę klasy `CGridView` i określamy ścieżkę dostępu. Drugim parametrem jest tablica, w której ustawiamy dostawcę danych `dataProvider`. W ten sposób uzyskujemy dostęp do danych, którymi zostanie wypełniona tabela wygenerowana przez system (listing 8.11).

**Listing 8.11.** Widok, w którym zastosowano wtyczkę CGridView

```

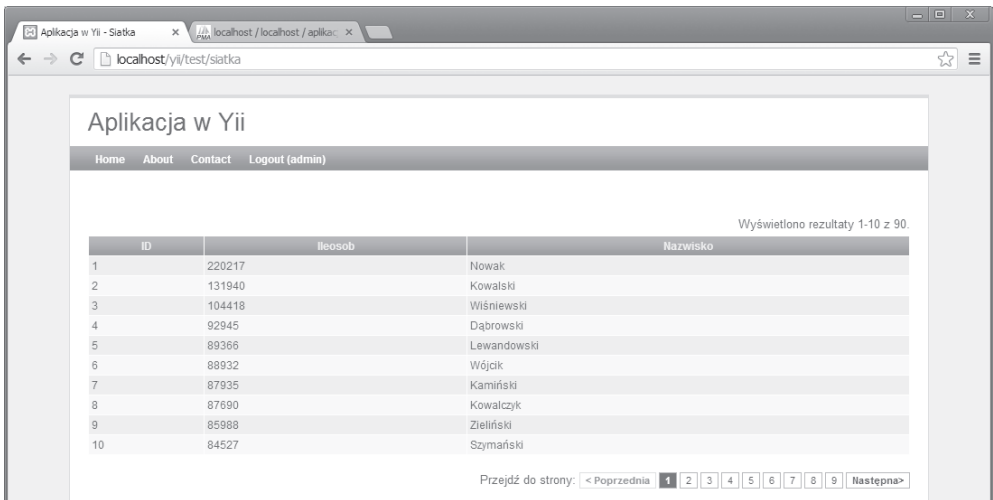
<?php

$this->widget('zii.widgets.grid.CGridView', array(
    'dataProvider'=>$Dane,
));

?>

```

Gotowe! Wywołajmy teraz w przeglądarce internetowej stronę *http://localhost/yii/test/siatka*. Aplikacja powinna pobrać z bazy dane i wyświetlić je w układzie tabeli. Aby posortować dane, wystarczy kliknąć nagłówek odpowiedniej kolumny. Powinno też zauważyć, że dane zostały rozdzielone na kilka kolejnych stron (rysunek 8.2).



**Rysunek 8.2.** Tak wygląda tabela wygenerowana przez aplikację. Wyświetlone dane pochodzą z bazy danych

# Skorowidz

## A

- action, 20
- actionAdmin(), 122
- actionCreate(), 121
- actionDelete(), 122
- actionIndex(), 24, 29, 50, 122, 132
- actionUpdate(), 121
- actionView(), 120
- ActiveRecord, 32, 42, 43, 145, 254
- addslashes(), 29
- AdminController, 170
- akcja
  - parametry, 21
- Apache, 10
- aplikacja
  - debugowanie, 251
- applyOrder(), 128
- arkusz stylów, 155, 201
- attributeLabels(), 48
- attributeName, 75
- attributes, 51

## B

- basePath, 87
- baza danych, 23
  - tworzenie, 23
- biblioteki
  - ładowanie, 88
- bindParam(), 29
- błędy aplikacji, 90
- button, 64

## C

- CActiveForm, 47
- CActiveRecord, 32, 48
- CAPTCHA, 130, 131, 133
- CBreadcrumbs, 141
- CController, 20
- CDbConnection, 30
- CDetailView, 140
- CFileLogRoute, 93
- CGridView, 143
- charset, 23
- checkBox(), 59
- CHtml, 64
- ciągi znakowe, 77
  - dozwolone, 77
- class, 30, 90
- className, 75
- CListView, 138
- CMenu, 137
- CodeIgniter, 17, 82, 257
- compare, 71
- compareValue, 71
- components, 22
- confirm, 97
- connectionString, 23
- Controller, 19
- count(), 36, 41
- countBySql(), 42
- createCommand(), 24
- CRUD, 118, 119, 124
- CSort(), 128
- czas, 73
  - wzorce, 74

**D**

dane konfiguracyjne, 100  
 DAO, 24  
 data, 73  
   wzorce, 74  
 date, 73  
 db, 23  
 dbname, 23  
 debugger, debugowanie, 251  
   opcje, 255  
 deklaracji action, 20  
 DELETE, 25  
 delete(), 39  
 deleteAll(), 40  
 dodawanie ścieżki dostępu  
   Linux, 14  
   Windows, 13  
 dropDownList(), 56  
 dzielenie na strony, 125  
 dzienniki systemowe, 90

**E**

emails, 92  
 enctype, 62  
 endWidget(), 49  
 error(), 70  
 errorSummary(), 49, 70  
 execute(), 25, 26  
 exploit, 17

**F**

fileField(), 61  
 filter, 76  
 filters(), 119  
 find(), 33, 39  
 findAll(), 36  
 findByAttributes(), 33, 35  
 findByPk(), 33, 37  
 findByPK(), 34  
 findBySql(), 33, 36  
 format, 73  
 format daty i czasu, 73  
 formularz, 47  
   budowa, 47  
   elementy, 52  
   tworzenie, 116  
 funkcja  
   addslashes(), 29  
   count(), 36

**G**

getFlash(), 104  
 Gii, 52, 109  
   logowanie, 111  
   uruchomienie, 109

**H**

hashFlash(), 104  
 hiddenField(), 63  
 host, 23  
 htmlButton(), 64  
 htmlOptions, 62

**I**

Index, 20  
 INSERT, 25

**K**

katalog  
   protected/models, 32, 48  
   protected/views, 45, 49  
 katalogu protected\config, 16  
 katalogu protected/config, 21, 22, 30, 47, 87, 109  
 katalogu protected/controllers, 26, 50  
 katalogu protected/log, 93  
 katalogu protected\controllers, 20  
 katalogu protected\views\layouts, 45  
 katalogu upload, 136  
 katalogu yii, 14  
 klasa  
    CActiveRecord, 48  
 klucz  
   attributeName, 75  
   basePath, 87  
   class, 30, 90  
   className, 75  
   compareValue, 71  
   components, 22  
   connectionString, 23  
   db, 23  
   emails, 92  
   filter, 76  
   format, 73  
   gii, 109  
   language, 47  
   levels, 90  
   options, 57  
   params, 100  
   password, 23



- preload, 90
- routes, 90
- type, 80
- username, 23
- value, 74
- komponenty
  - ładowanie, 88
- komunikaty
  - dodawanie, 103
  - poziomy, 92
    - error, 93
    - info, 93
    - profile, 93
    - trace, 93
    - warning, 93
  - w sesji, 103
- kontroler, 19, 50
  - tworzenie, 114
- konwertowanie otrzymanych danych, 76

**L**

- labelEx(), 49
- levels, 90
- liczba wpisanych znaków, 78
  - dozwolona, 78
- loadModel(), 123
- logPath, 93

**Ł**

- ładowanie
  - komponentów, 88
- ładowanie bibliotek, 88
- łącza
  - tworzenie, 94

**M**

- metoda
  - actionAdmin(), 122
  - actionCreate(), 121
  - actionDelete(), 122
  - actionIndex(), 20, 24, 29, 50, 122, 132
  - actionUpdate(), 121
  - actionView(), 120
  - attributeLabels(), 48
  - beginWidget(), 49
  - bindParam(), 29
  - checkBox(), 59
  - count(), 41
  - createCommand(), 24
  - delete(), 39

- deleteAll(), 40
- dropDownList(), 56
- endWidget(), 49
- error(), 70
- errorSummary(), 49, 70
- execute(), 25, 26
- fileField(), 61
- filters(), 119
- find(), 33, 39
- findAll(), 36
- findByAttributes(), 33, 35
- findByPk(), 33, 34, 37
- findBySql(), 33, 36
- getFlash(), 104
- hashFlash(), 104
- hiddenField(), 63
- htmlButton(), 64
- labelEx(), 49
- loadModel(), 123
- model(), 34
- passwordField(), 54
- performAjaxValidation(), 123
- POST, 29
- query(), 24-27
- queryColumn(), 27
- queryRow(), 27
- queryScalar(), 27
- radioButton(), 58
- render(), 51
- resetButton(), 67
- rules(), 68
- save(), 33, 37
- setAttributes(), 132
- submitButton(), 49, 65
- textArea(), 55
- textField(), 52
- updateAll(), 39
- updateByPk(), 38
- validate(), 51
- widget(), 127, 133, 138
- mod\_rewrite, 16
- model, 19, 22, 48
  - tworzenie, 112
- model(), 34
- multipart, 62
- MVC, 19
- MySQL, 10

**O**

- operator porównania, 72
  - !=, 72
  - <, 72
  - <=, 72

operator porównania  
 =, 72  
 ==, 72  
 >, 72  
 >=, 72  
 options, 57

**P**

params, 100  
 password, 23  
 passwordField(), 54  
 Path, 13  
 performAjaxValidation(), 123  
 PHP, 10  
 phpMyAdmin, 23  
 plik  
 .htaccess, 16, 94, 154, 201  
 aplikacji, 153, 200  
 index.php, 16, 98  
 konfiguracyjny, 153, 200  
 main.php, 16, 17, 21, 22, 30, 47, 87, 100, 109  
 php.ini, 10  
 szablonu strony, 155, 202  
 pole  
 checkbox, 59  
 file, 61  
 hidden, 63  
 listy rozwijanej, 56  
 opcji (przełącznik), 58  
 password, 54  
 puste, 70  
 radio, 58  
 select, 56  
 selektor plików, 61  
 tekstowe, 52  
 text, 52  
 textarea, 55  
 ukryte, 63  
 wartość domyślna, 74  
 wyboru, 59  
 z ukrytą treścią, 54  
 poprawność adresu e-mail, 73  
 porównywanie, 71  
 POST, 29  
 poziomy komunikatów, 93  
 preload, 90  
 przekierowania, 103  
 przycisk, 64  
 button, 64  
 reset, 67  
 submit, 65  
 przycisk przywracania danych domyślnych formularza, 67

przycisk wysłania formularza, 65  
 puste pola, 70

**Q**

Query Builder, 43  
 query(), 24, 25, 26  
 queryAll(), 27  
 queryColumn(), 27  
 queryRow(), 27  
 queryScalar(), 27

**R**

radioButton(), 58  
 range, 77  
 render(), 51  
 required, 70  
 reset, 67  
 resetButton(), 67  
 routes, 90  
 rules(), 68

**S**

save(), 33, 37  
 SELECT, 25  
 selected, 57  
 setAttributes(), 132  
 setFlash(), 104  
 sortowanie danych, 128  
 SQL, 43, 199, 254  
 strony statyczne, 101  
 strtoupper(), 76  
 submit, 65  
 submitButton(), 49, 65  
 aystem Yii, 9  
 szablony, 105  
 korzystanie, 105

**Ś**

ścieżka dostępu, 99  
 zmiana, 99

**T**

textArea(), 55  
 textField(), 49, 52  
 typ zmiennej, 81  
 sprawdzanie, 80  
 type, 80  
 tytuł, 87  
 zmiana, 87

**U**

unique, 81  
 UPDATE, 25  
 updateAll(), 39  
 updateByPk(), 38  
 URL, 81  
   walidacja, 81  
 urlManager, 16, 21  
 username, 23  
 utf8, 23

**V**

validate(), 51  
 value, 74  
 VARCHAR, 24

**W**

walidacja adresu URL, 81  
 walidacja danych, 68  
 wartości liczb, 79  
 wartość  
   date, 81  
   datetime, 81  
   float, 81  
   integer, 81  
   pola  
     sprawdzanie, 82  
   string, 81  
   time, 81  
   unikalna, 81  
 widget(), 127, 133, 138  
 widok, 19, 43, 49  
 wielowierszowe pole tekstowe, 55  
 wzorce sprawdzania formatu daty i czasu, 74

wzorzec  
 ?, 74  
 a, 74  
 d, 74  
 dd, 74  
 h, 74  
 hh, 74  
 m, 74  
 mm, 74  
 MM, 74  
 MMMM, 74  
 model – widok – kontroler, 19  
 s, 74  
 ss, 74  
 yy, 74  
 yyyy, 74

**X**

XAMPP, 10, 23

**Y**

Yii, 9, 10, 17, 19, 257  
   instalacja, 13  
   możliwości, 125  
   wymagania, 11

**Z**

zapisywanie plików na serwerze, 134  
 zapytanie  
   DELETE, 25  
   INSERT, 25  
   UPDATE, 25  
   SELECT, 25  
 Zii, 137  
   CBreadcrumbs, 141  
   CDetailView, 140  
   CGridView, 143  
   CListView, 138  
   CMenu, 137  
   rozszerzenia, 137  
 zmiana tytułu, 87



# 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**

# yiiframework

Yii (akronim od „Yes, it is!”) to fantastyczny framework oparty na PHP, przeznaczony do tworzenia profesjonalnych aplikacji sieciowych wielkiej skali. Świetnie nadaje się do budowy interaktywnych witryn, których twórcy przewidują dużą liczbę odwiedzin, ponieważ jest nie tylko lekki, ale także wyposażony w mechanizmy zapobiegające ładowaniu się i inicjalizacji dodatkowych bibliotek aż do momentu wywołania metody z ich zasobu. Pozwala także stworzyć stronę WWW z wyraźnie oddzielonymi od siebie sferami działania, według struktury model-widok-kontroler. Oferuje mnóstwo funkcji, po które można sięgnąć w bardzo prosty sposób, bez potrzeby zgłębiania kodu plików zawierających metody wykonujące te funkcje w systemie.

W tej książce znajdziesz wszystkie ważne informacje dotyczące frameworka Yii, od sposobu instalacji aż po sprawdzanie poprawności danych gotowej aplikacji, wprowadzanych poprzez formularz. Dowiesz się, jak w praktyce wykorzystać strukturę model-widok-kontroler, stworzyć formularz i skonfigurować zaawansowane opcje frameworka. Poznasz narzędzia do automatycznego generowania kodu (Gii) i interesującą, bardzo przydatną bibliotekę rozszerzeń (Zii). Nauczysz się budować blog z wykorzystaniem ActiveRecord oraz SQL, a także usuwać błędy z Twojej aplikacji. Z tą książką w pełni opanujesz Yii!

- Pobieranie i instalacja systemu Yii
- Model-widok-kontroler
- Tworzenie formularzy
- Zaawansowana konfiguracja
- Gii
- Zaawansowane możliwości
- Zii
- Budowa bloga — ActiveRecord i SQL
- Odpluskwanie aplikacji

## Yii — genialne narzędzie na miarę Twoich potrzeb!

**helion.pl**  
księgarnia internetowa

Nr katalogowy: 14635

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

Zamówienia telefoniczne:  
**0 801 339900**  
**0 601 339900**



**Helion**

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

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

sięgnij po WIĘCEJ



KOD KORZYŚCI

ISBN 978-83-246-7920-1



9 788324 679201

Cena: 39,90 zł

Informatyka w najlepszym wydaniu