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?

Helion

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
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

Printed in Poland.

Spis treści

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 tworzyć 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 tablice 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, uzyskamy 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 uchwytem 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*[1].

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()
   {
```

[1] *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 potrzebnymi parametrami. Pierwszy z tych parametrów to nazwa kontrolera i metody, a kolejnymi są zmienne wraz z wartościami (listing 8.9).

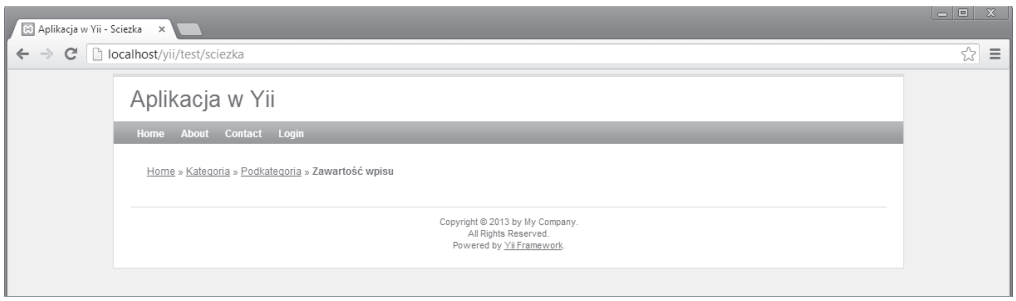
Listing 8.9. *Widok z definicją łączy wskazujących pozycję bieżącej strony w witrynie*

```
<?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).

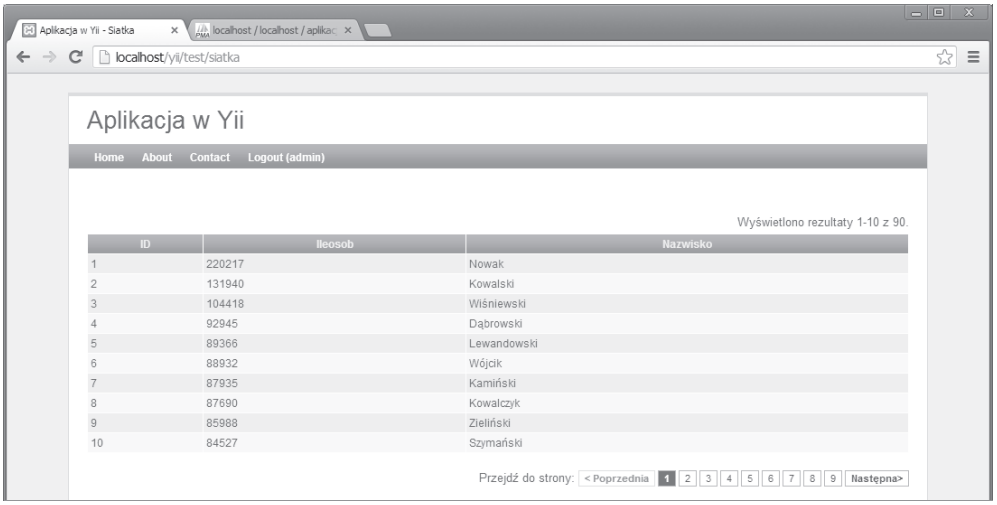
Listing 8.11. *Widok, w którym zastosowano wtyczkę CGridView*

```
<?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. Powinieneś 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

D

E

F

G

H

I

K

L

Ł

M

O

U

V

W

X

Y

Z

yiframework

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**
- **Odpluskwianie aplikacji**

Yii — genialne narzędzie na miarę Twoich potrzeb!

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

ISBN 978-83-246-7920-1

Cena: 39,90 zł