



Technologia i rozwiązania

Projektowanie systemów CMS przy użyciu PHP i jQuery

Zbuduj CMS na miarę swoich potrzeb!

- Jak zaprojektować jądro systemu CMS?
- Jak zarządzać użytkownikami?
- Jak przygotować instalator?



Kae Verens



Tytuł oryginału: CMS Design Using PHP and jQuery

Tłumaczenie: Łukasz Piwko

ISBN: 978-83-246-3365-4

Copyright © Packt Publishing 2010. First published in the English language under the title:
“CMS Design Using PHP and jQuery”

Polish language edition published by Helion S.A.

Copyright © 2011

All rights reserved

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without permission in writing from the publisher, except in the case of brief quotations embedded in critical articles or reviews.

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.

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

Materiały graficzne na okładce zostały wykorzystane za zgodą iStockPhoto Inc.

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/psycms>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Printed in Poland.

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

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

Spis treści

O autorze	7
Podziękowania	9
O recenzentach	11
Wstęp	13
Rozdział 1. Projekt jądra systemu CMS	17
Sekcje prywatna i publiczna systemu CMS	18
Front systemu CMS	18
Panel administracji	20
Wtyczki	21
Pliki i bazy danych	22
Struktura katalogów	22
Struktura bazy danych	24
Plik konfiguracyjny	25
Witaj, świecie	26
Konfiguracja	26
Kontroler frontu	29
Wczytywanie danych strony z bazy danych	31
Podsumowanie	39
Rozdział 2. Zarządzanie użytkownikami	41
Rodzaje użytkowników	41
Role	42
Tabele bazy danych	44

Strona logowania do panelu administracyjnego	46
Logowanie	54
Wylogowywanie	60
Odzyskiwanie hasła	62
Zarządzanie użytkownikami	66
Usuwanie użytkowników	68
Tworzenie i modyfikowanie użytkowników	69
Podsumowanie	72
Rozdział 3. Zarządzanie stronami — część pierwsza	73
Strony w systemie CMS	73
Wyświetlanie listy stron w panelu administracyjnym	74
Widok hierarchii stron	77
Aranżacja hierarchii stron	81
Administracja stronami	82
Asynchroniczne wypełnianie listy rodziców	90
Podsumowanie	92
Rozdział 4. Zarządzanie stronami — część druga	93
Daty	93
Zapisywanie stron	96
Tworzenie stron najwyższego poziomu	99
Tworzenie podstron	101
Usuwanie stron	102
Edycja tekstu sformatowanego przy użyciu narzędzia CKEditor	104
Zarządzanie plikami za pomocą narzędzia KFM	107
Podsumowanie	113
Rozdział 5. Szablony — część pierwsza	115
Motywy i szablony	116
Struktura plików motywu	118
Konfiguracja systemu Smarty	120
Frontowe menu nawigacyjne	125
Podsumowanie	131
Rozdział 6. Szablony — część druga	133
Dodawanie jQuery do menu	133
Przygotowanie menu Filament Group	134
Integracja menu	137
Ustawianie motywów w panelu administracyjnym	140
Wybór szablonu strony w panelu administracyjnym	146
Smarty w treści stron	149
Podsumowanie	151

Rozdział 7. Wtyczki	153
Co to są wtyczki?	153
Zdarzenia w systemie CMS	154
Typy stron	155
Sekcje w panelu administracyjnym	155
Dodatki do wszystkich stron w panelu administracyjnym	156
Przykład konfiguracji wtyczki	156
Włączanie wtyczek	158
Obsługa uaktualnień i tabel bazy danych	162
Własne menu użytkownika w panelu administracyjnym	165
Dodawanie zdarzeń do systemu CMS	172
Dodawanie zakładki do panelu administracji	178
Podsumowanie	184
Rozdział 8. Wtyczka do tworzenia formularzy	185
Jak to ma działać	185
Definicja wtyczki	186
Typy stron w panelu administracyjnym	188
Dodawanie formularzy do panelu administracyjnego	191
Definiowanie pól formularza	197
Wyświetlanie formularza na froncie	202
Skrypt obsługi wysyłania formularza	207
Wysyłanie wiadomości pocztą elektroniczną	209
Zapisywanie danych formularza w bazie danych	211
Eksport zapisanych danych	212
Podsumowanie	214
Rozdział 9. Wtyczka galerii obrazów	215
Konfiguracja wtyczki	216
Zakładki w panelu administracyjnym	217
Ustawienia początkowe	218
Wysyłanie obrazów	220
Obsługa wysyłania plików	222
Reguła mod_rewrite	223
Usuwanie obrazów	224
Frontowy widok galerii	225
Karta ustawień	229
Galeria siatkowa	232
Podsumowanie	236
Rozdział 10. Okienka i widżety — część pierwsza	237
Wtyczka do tworzenia okienek	238
Rejestracja okienka	240
Administracja okienkami	242
Wyświetlanie okienek	243

Tworzenie wtyczki fragmentów treści	246
Wstawianie widżetów do okienek	247
Wyświetlanie widżetów	248
Przeciąganie widżetów do okienek	249
Zapisywanie zawartości okienka	252
Wyświetlanie okienek na froncie	255
Podsumowanie	257
Rozdział 11. Okienka i widżety — część druga	259
Formularze do konfiguracji widżetów	259
Zapisywanie treści fragmentu	265
Zmianie nazw widżetów	267
Widoczność nagłówek widżetów	268
Wyłączanie widżetów	269
Wyłączanie okienek	271
Usuwanie okienek	273
Widoczność okienek na stronach — kod od strony administracyjnej	274
Widoczność okienek na stronach — kod od strony frontowej	278
Widoczność widżetów na stronach	279
Podsumowanie	281
Rozdział 12. Budowa instalatora	283
Instalacja maszyny wirtualnej	284
Instalacja narzędzia WMware Player	284
Instalacja maszyny wirtualnej	284
Instalacja CMS-a w maszynie wirtualnej	287
Tworzenie instalatora	290
Zmiany w jądrze CMS-a	290
Instalator	291
Sprawdzanie, czego brakuje	292
Dane konfiguracyjne	296
Podsumowanie	302
Skorowidz	303

Zarządzanie stronami — część pierwsza

W tym rozdziale stworzymy formularz do zarządzania stronami i zbudujemy system pozwalający przenosić strony metodą przeciągania.

Omówione zostaną następujące tematy:

- Żądanie i generowanie stron
- Wyświetlanie listy stron w panelu administracyjnym
- Administracja stronami

System zarządzania stronami zostanie ukończony dopiero w kolejnym rozdziale, w którym opiszę zapisywanie stron w bazie danych oraz dodam edytor tekstu i menedżera plików.

Strony w systemie CMS

Zgodnie z tym, co napisałem w rozdziale 1., strona to zasadniczo treść, która powinna zostać wyświetlona w odpowiedzi na żądanie określonego adresu URL.

W serwisach nieopartych na systemach CMS pojęcie strony jest bardzo wyraźne, ponieważ każdy adres URL zwraca konkretny plik HTML. Natomiast w systemie CMS strony są generowane dynamicznie i mogą być wzbogacone o rozmaite wtyczki, rodzaje prezentacji zależne od tego, czy użytkownik czegoś szuka, czy zastosowano opcję podziału na podstrony, itd.

W większości witryn internetowych stronę można w uproszczeniu zdefiniować jako duży obszar treści pośrodku okna przeglądarki. Czasami jednak trudno zaakceptować taką definicję, gdyż to, co widać na ekranie, może w istocie być składanką fragmentów treści z różnych miejsc serwisu.

Problem tych różnic rozwiązujemy poprzez zastosowanie „typów” stron, z których każdy może być zaprezentowany w inny sposób. Wśród typów stron można wymienić galerie, formularze, strony z wiadomościami, wyniki wyszukiwania itd.

W tym rozdziale pokażę, jak utworzyć najprostszy typ strony, o nazwie *normalny*. W panelu administracyjnym formularz służący do tworzenia tego typu stron będzie zawierał duży obszar tekstowy na treść, która po zatwierdzeniu będzie wyświetlana na stronie. Można zastosować też inną nazwę, taką jak np. „domyślny”, ale ponieważ z CMS-ów często korzystają też osoby niemające wiedzy technicznej, lepiej zastosować słowo, którego znaczenie jest raczej oczywiste. Niejednokrotnie klienci pytali mnie, co oznacza słowo „default” (domyślny), natomiast słowo „normal” (normalny) jeszcze nikogo nie dziwiło.

Jak zapewne pamiętasz, w rozdziale 1. napisałem, co powinno wchodzić w skład jądra systemu, a co należałoby udostępnić w postaci wtyczki.

Każdy system CMS powinien umożliwiać tworzenie przynajmniej jednego najprostszego rodzaju stron. Dlatego typ normalny będzie wbudowany w jądro, a nie udostępniany jako dodatek.

Wyświetlanie listy stron w panelu administracyjnym

Zacniemy od dodania odnośnika *Strony* do menu administracyjnego. W tym celu otwórz plik */ww.admin/header.php* i dodaj do niego poniższy pogrubiony wiersz kodu:

```
<ul>
  <li><a href="/ww.admin/pages.php">Strony</a></li>
  <li><a href="/ww.admin/users.php">Użytkownicy</a></li>
```

Jeszcze jedno: po zalogowaniu do sekcji administracyjnej użytkownik powinien być automatycznie przenoszony do sekcji zarządzania stronami, gdyż większość czynności związanych z zarządzaniem systemem dotyczy właśnie stron serwisu.

W związku z tym zmienimy zawartość pliku */ww.admin/index.php*, aby zawierał ten sam kod, co plik */ww.admin/pages.php*. Zamień zawartość pliku */ww.admin/index.php* na następujący kod:

```
<?php
require 'pages.php';
```

Teraz zajmiemy się budową sekcji *Strony*.

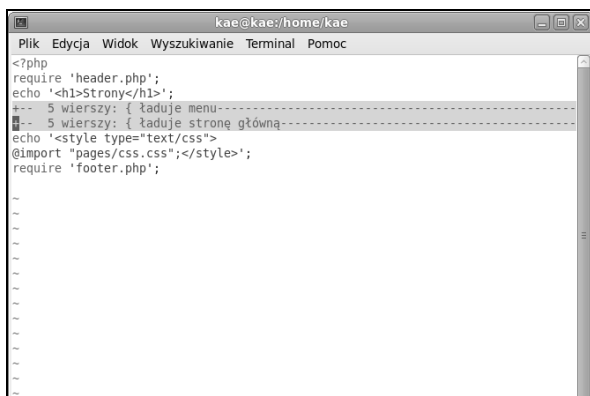
Zacniemy od utworzenia pliku `/ww.admin/pages.php`:

```
<?php
require 'header.php';
echo '<h1>Pages</h1>';
// { ładuje menu
echo '<div class="left-menu">';
require 'pages/menu.php';
echo '</div>';
// }
// { ładuje stronę główną
echo '<div class="has-left-menu">';
require 'pages/forms.php';
echo '</div>';
// }
echo '<style type="text/css">
    @import "pages/css.css";</style>';
require 'footer.php';
```

Zwróć uwagę na specjalne komentarze, w które ująłem bloki kodu (początek bloku oznaczony jest znakami `//{`, a koniec znakami `//}`).

Komentarze te dodałem dlatego, gdyż niektóre edytory oferują tzw. **funkcję zwijania**, która pozwala zwijać bloki kodu ujęte w specjalne znaczniki, tak aby widoczny był tylko pierwszy wiersz.

Przykładowo u mnie w edytorze Vim powyższy fragment kodu wyświetlony jest tak, jak widać na rysunku:



The screenshot shows a Vim editor window titled 'kae@kae:/home/kae'. The menu bar includes 'Plik', 'Edycja', 'Widok', 'Wyszukiwanie', 'Terminal', and 'Pomoc'. The code is displayed with fold markers: a double-dash followed by a vertical bar and a number (e.g., '-- 5 wierszy: { ładuje menu-----'). The code content is the same as in the previous block, but with the fold markers inserted at the beginning of the comment lines.

Zadaniem skryptu zapisanego w pliku `pages.php` jest wczytanie nagłówek, menu, formularza i stopki. W dalszej części rozdziału jeszcze do niego wrócimy.

Teraz utwórz katalog `/ww.admin/pages`, a w nim stwórz plik o nazwie `/ww.admin/pages/forms.php`:

```
<h2>TU BĘDZIE FORMULARZ</h2>
```

Możemy przejść do tworzenia menu strony. Utwórz plik `/ww.admin/pages/menu.php` o następującej treści:

```
<?php
echo '<div id="pages-wrapper">';
$rs=dbAll('select id,type,name,parent from pages order by ord,name');
$pages=array();
foreach($rs as $r){
    if(!isset($pages[$r['parent']]))$pages[$r['parent']]=array();
    $pages[$r['parent']][]=$r;
}
function show_pages($id,$pages){
    if(!isset($pages[$id]))return;
    echo '<ul>';
    foreach($pages[$id] as $page){
        echo '<li id="page_'.$page['id'].'">'
            . '<a href="pages.php?id='.$page['id'].'">'
            . '<ins>&nbsp;</ins>'.htmlspecialchars($page['name'])
            . '</a>';
        show_pages($page['id'],$pages);
        echo '</li>';
    }
    echo '</ul>';
}
show_pages(0,$pages);
echo '</div>';
```

Skrypt ten generuje nieuporządkowaną listę stron.

Zwróć uwagę na użycie pola `parent`. W większości witryn strony są rozmieszczone w strukturze hierarchicznej z relacjami typu „rodzic – dziecko”. Każda strona jest „dzieckiem” jakiejś innej strony lub „korzenia” witryny. W polu `parent` zapisany jest identyfikator strony będącej nadrzędną wobec danej strony.

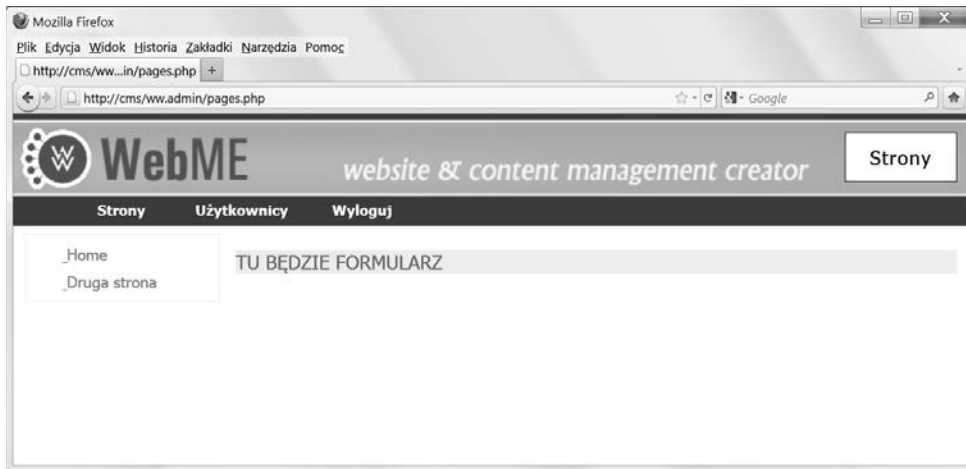
Stronę główną (tzn. tę, która zostanie wyświetlona po wpisaniu w przeglądarce adresu `http://cms/` bez wskazania konkretnej strony) można wyznaczyć na dwa sposoby.

1. Tworząc w bazie danych jedną stronę z polem `parent` o wartości 0, oznaczającej, że ta strona nie ma rodzica — to ta strona zostanie wyświetlona po wpisaniu w oknie przeglądarki adresu `http://cms/`. Wówczas wszystkie strony typu `http://cms/nazwastrony` będą miały w polu `parent` identyfikator strony, która ma w tym polu wartość 0.
2. Tworząc wiele stron z polem `parent` o wartości 0, z których każda będzie tzw. stroną najwyższego poziomu. Jedna z nich musiałaby wówczas mieć specjalną wartość w polu `special`, wskazującą, że to jest strona główna. W takim przypadku strony typu `http://cms/nazwastrony` będą miały rodzica 0, a strona `http://cms/` może być przechowywana w dowolnym miejscu bazy danych.

Pierwsze rozwiązanie ma jedną poważną wadę: aby zmienić stronę główną, trzeba przenieść bieżącą stronę główną pod jakąś inną stronę (albo ją usunąć), a następnie przesunąć wszystkie strony potomne bieżącej strony głównej tak, aby miały w polu parent identyfikator nowej strony głównej. Może to być bardzo skomplikowane, jeśli nowa strona główna ma już jakieś podstrony — zwłaszcza jeśli nazwy niektórych się powtarzają.

Drugie rozwiązanie jest o wiele lepsze, ponieważ pozwala na bezproblemową zmianę strony głównej.

Teraz nasz serwis wygląda tak, jak widać na poniższym rysunku (do budowy użyte zostały strony wprowadzone do bazy w rozdziale 1.).



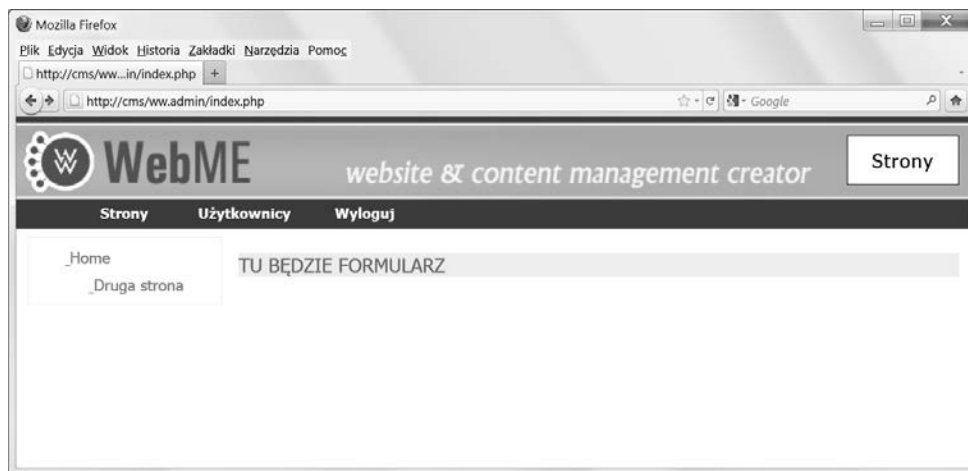
Widok hierarchii stron

Zmodyfikujemy nieco bazę danych, aby można było wyświetlić hierarchię stron.

W tym celu otwórz konsolę MySQL i zmień definicję drugiej strony tak, aby w jej polu parent znajdował się identyfikator strony głównej:

```
mysql> select id,name,parent from pages;
+----+-----+-----+
| id | name      | parent |
+----+-----+-----+
| 24 | Home      | 0      |
| 25 | Druga strona | 0      |
+----+-----+-----+
2 rows in set (0.00 sec)
mysql> update pages set parent=24 where id=25;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

Po dokonaniu zmian odśwież stronę w przeglądarce:



Jak widać, wpis **drugiej strony** jest nieco wcięty, gdyż jest ona potomkiem **strony głównej** i w kodzie HTML znajduje się w zagnieżdżonym elemencie `ul`.

Widok ten możemy poprawić.

Istnieje wtyczka jQuery o nazwie `jstree`, która przetwarza drzewa stworzone przy użyciu elementów `ul` w interfejsy podobne do interfejsów menedżerów plików.

Ponadto pozwala ona na przeciąganie węzłów drzewa i wiązanie zdarzeń z kliknięciami tych węzłów.

Skorzystamy z tych funkcji później, kiedy będziemy pracować nad mechanizmem tworzenia i usuwania stron oraz zmiany hierarchii stron poprzez ich przeciąganie.

Utwórz w katalogu głównym witryny folder o nazwie `/j/`.

Przypomnę, że w rozdziale 1. została podjęta decyzja, iż wszystkie nazwy katalogów w systemie, które są dłuższe niż dwa znaki, będą zawierać kropkę.

Powodem, dla którego katalog ten nazwałem `/j/` zamiast np. `/ww.javascript/`, jest mała liczba znaków pozwalająca zaoszczędzić kilka bajtów transferu. To może być ważne, jeśli z serwisu będą korzystać użytkownicy dysponujący łączami o niskich parametrach, takich jakie są np. w smartfonach.

Skrócenie jednej nazwy może na niewiele się zda, ale jeśli wyrobisz sobie nawyk stosowania takich krótkich nazw, to skumulowany efekt pozwoli zaoszczędzić sekundę lub dwie przy pobieraniu stron.

Jedna drobna optymalizacja nie ma znaczenia, ale połączenie dużej liczby takich optymalizacji to już całkiem co innego.

W każdym bądź razie stworzymy folder `/j/` i pobieramy skrypt `jstree` ze strony <http://jstree.com> oraz wypakowujemy zawartość archiwum tak, aby ścieżka do pliku `jquery.tree.js` była następująca: `/j/jquery.jstree/jquery.tree.js`.

Do budowy tego CMS-a użyta została wersja 0.9.9a skryptu.

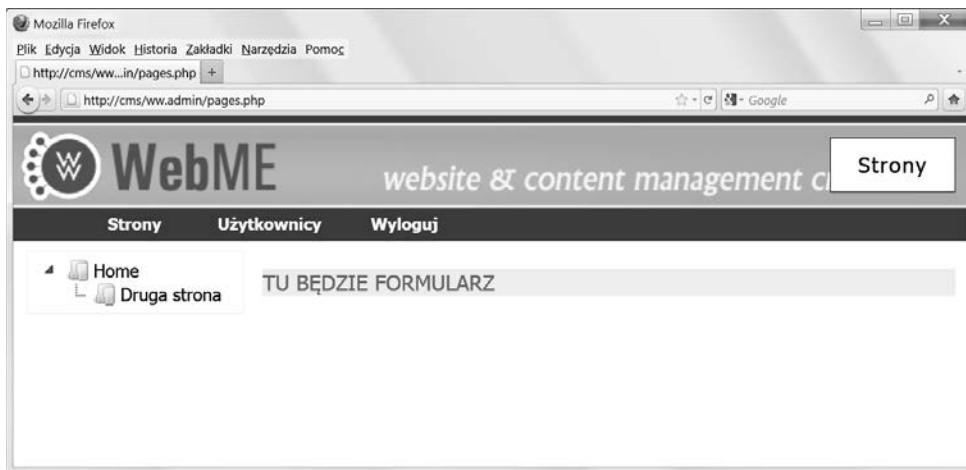
Teraz otwórz plik `/ww.admin/pages/menu.php` i dodaj do niego poniższe wiersze kodu zaznaczone pogrubieniem.

```
<script src="/j/jquery.jstree/jquery.tree.js"></script>
<script src="/ww.admin/pages/menu.js"></script>
<?php
```

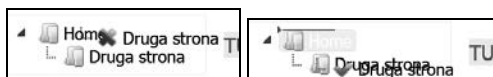
Następnie utwórz plik `/ww.admin/pages/menu.js`:

```
$(function(){
    $('#pages-wrapper').tree();
});
```

Od razu na ekranie pojawi nam się piękne drzewo reprezentujące hierarchię stron serwisu, jak pokazano na poniższym rysunku:



Nazwy stron można przeciągać w różne miejsca, które w czasie operacji są zaznaczane specjalnym symbolem, tak jak widać na dwóch kolejnych zrzutach ekranu:



Zanim przejdziemy do rzeczywistej pracy nad stronami, dokonamy jeszcze jednej poprawki tego menu. Stworzymy przycisk pozwalający dodawać strony najwyższego poziomu oraz zaimplementujemy funkcję zapamiętywania zdarzeń przeciągania stron, aby miały one trwały skutek.

Otwórz plik `/ww.admin/pages/menu.js` i zmień jego zawartość na następującą:

```
$(function(){
  $('#pages-wrapper').tree({
    callback:{
      onchange:function(node,tree){
        document.location='pages.php?action=edit&id='
          +node.id.replace(/.*_/,'');
      },
      onmove:function(node){
        var p=$.tree.focused().parent(node);
        var new_order=[],nodes=node.parentNode.childNodes;
        for(var i=0;i<nodes.length;++i)
          new_order.push(nodes[i].id.replace(/.*_/,''));
        $.getJSON('/ww.admin/pages/move_page.php?id='
          +node.id.replace(/.*_/,'')+'&parent_id='
          +(p==-1?0:p[0].id.replace(/.*_/,''))
          +'&order='+new_order);
      }
    }
  });
  var div=$(
    '<div><i>Prawy przycisk myszy wyświetla opcje</i><br /><br /></div>');
  $('<button>Dodaj stronę główną</button>')
    .click(pages_add_main_page)
    .appendTo(div);
  div.appendTo('#pages-wrapper');
});
function pages_add_main_page(){
```

Za pomocą tego kodu wzbogaciliśmy drzewo stron o kilka funkcji.

Po pierwsze, została dodana funkcja zwrotna `onchange`.

Kliknięcie węzła drzewa (nazwy jednej ze stron) powoduje przekierowanie przeglądarki na stronę `pages.php?edit=` z identyfikatorem klikniętej strony na końcu. Podczas tworzenia elementu `ul` reprezentującego drzewo dla każdego podelementu `li` zdefiniowaliśmy identyfikator w taki sposób, że nazwa strony o identyfikatorze 24 w bazie danych znajduje się na stronie w elemencie `li` o identyfikatorze `page_24`.

Zatem kiedy zostanie kliknięty jeden z węzłów (element `li`), musimy tylko usunąć część `page_` identyfikatora i resztę łańcucha użyć do otwarcia strony `pages.php`.

Po drugie, dodaliśmy funkcję zwrotną o nazwie `onmove`. Jej wywołanie następuje po zakończeniu operacji przeciągania elementu.

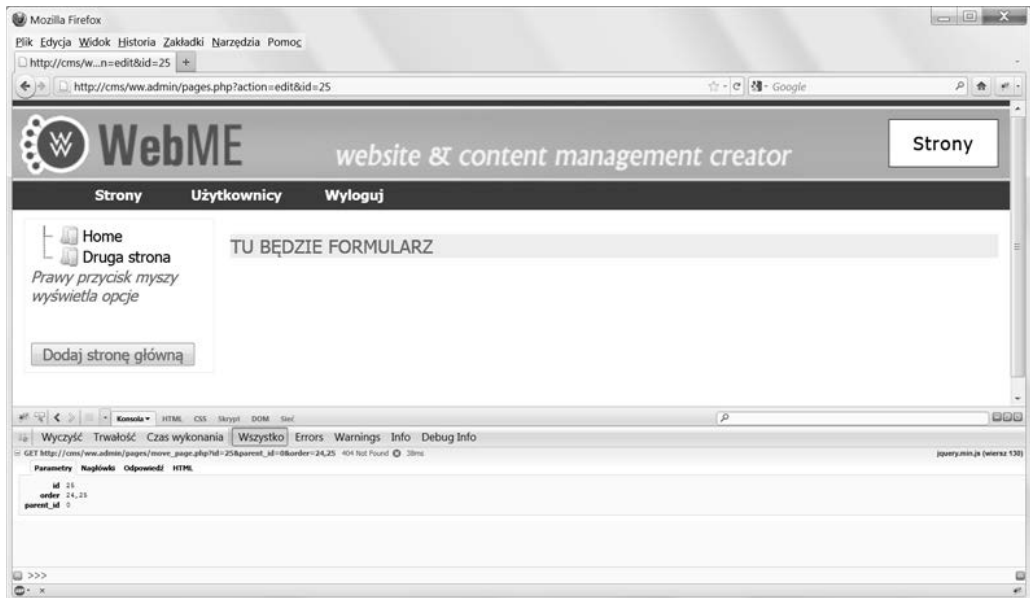
Jej treść jest nieco bardziej skomplikowana niż poprzedniej. Pobiera identyfikator nowego rodzica i tworzy tablicę, w której zapisuje identyfikatory wszystkich bezpośrednich podstron. Następnie wszystkie te dane wysyła do skryptu `/ww.admin/pages/move_page.php`, który niebawem stworzymy.

Na końcu dodaliśmy informację, że kliknięcie prawym przyciskiem myszy powoduje wyświetlenie dodatkowych opcji, którymi zajmiemy się później, oraz utworzyliśmy przycisk do tworzenia stron najwyższego poziomu, którego obsługą również zajmiemy się w dalszej części tego rozdziału. Aby kod ten nie powodował błędów, konieczne jest dodanie atrapy funkcji. Później zastąpimy ją prawdziwą definicją.

Aranżacja hierarchii stron

W tej chwili przeciągnięcie strony w inne miejsce na drzewie powoduje wykonanie wywołania Ajax do skryptu `/ww.admin/pages/move_page.php` i przekazanie do niego pewnych informacji.

Oto zrzut ekranu (widoczne okienko dodatku Firebug), na którym widać, jakie dane są przesyłane podczas takiej operacji przeciągania:



W tym wywołaniu zostały przesłane następujące informacje: identyfikator strony 25, identyfikator nowego rodzica 0 oraz nowa kolejność stron, których rodzic ma identyfikator 0 (25,24).

Czas na napisanie skryptu `/ww.admin/pages/move_page.php`:

```
<?php
require '../admin_libs.php';
$id=(int)$_REQUEST['id'];
$to=(int)$_REQUEST['parent_id'];
$order=explode(',',$_REQUEST['order']);
dbQuery('update pages set parent='.$to.' where id='.$id );
for($i=0;$i<count($order);++$i){
    $pid=(int)$order[$i];
    dbQuery("update pages set ord=$i where id=$pid");
    echo "update pages set ord=$i where id=$pid\n";
}
```

To proste! Ten skrypt zapisuje te informacje, które są przesyłane w wywołaniu Ajax.

Administracja stronami

Mamy już listę stron serwisu. Czas na dodanie funkcji umożliwiających ich modyfikowanie.

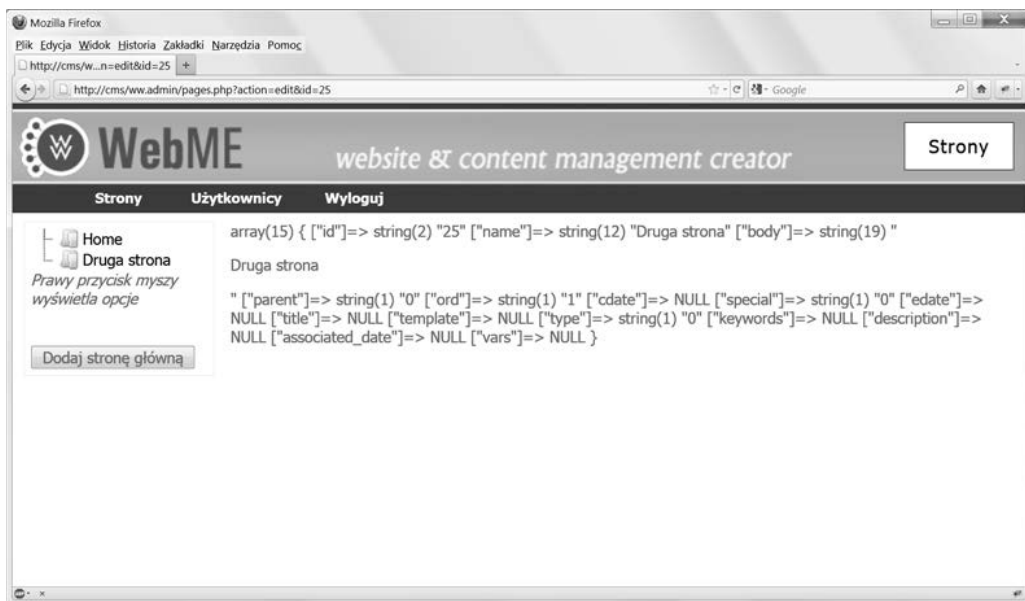
Kod formularza do tworzenia stron jest dość długi, dlatego będę go pokazywać i omawiać po kawałku. Otwórz plik `/ww.admin/pages/forms.php` i zastąp jego zawartość poniższym kodem:

```
<?php
if(isset($_REQUEST['id']))$id=(int)$_REQUEST['id'];
else $id=0;
if($id){ // sprawdzenie, czy strona o takim identyfikatorze istnieje
    $page=dbRow("SELECT * FROM pages WHERE id=$id");
    if($page!==false){
        $page_vars=json_decode($page['vars'],true);
        $edit=true;
    }
}
if(!isset($edit)){
    $parent=isset($_REQUEST['parent'])?
        (int)$_REQUEST['parent']:0;
    $special=0;
    if(isset($_REQUEST['hidden']))$special+=2;
    $page=array('parent'=>$parent,'type'=>'0','body'=>'',
        'name'=>,'title'=>,'ord'=>0,'description'=>,'
        'id'=>0,'keywords'=>,'special'=>$special,
        'template'=>);
    $page_vars=array();
    $id=0;
    $edit=false;
}
```


Kod ten inicjuje dwie tablice: `$page` do przechowywania najważniejszych informacji o stronie i `page_vars` na dane nienależące do głównej tabeli danych stron, np. informacje zapisane przez jakąś wtyczkę.

Jeśli w adresie URL zostanie wysłany identyfikator, to skrypt wczyta dane odpowiadającej mu strony.

Przykładowo: jeśli dodam do kodu wiersz `var_dump($page);`, a następnie wpiszę w przeglądarce adres `/ww.admin/pages.php?action=edit&id=25` (taka strona znajduje się w mojej bazie danych), to zobaczę następujący wynik:



Zostały wyświetlone wszystkie informacje o stronie, jakie znajdują się w tabeli bazy danych.

Gdyby w adresie URL został przekazany identyfikator 0 lub jakikolwiek inny, którego nie ma aktualnie w bazie danych, to tablica `$page` i tak zostałaby zainicjowana, ale pustymi wartościami.

Ponieważ strony mogą być bardzo skomplikowane, zwłaszcza gdy będzie wiele ich rodzajów dodanych przez wtyczki, podzielimy formularz na kilka zakładek.

Formularz tworzenia stron normalnego typu będzie się składał z dwóch zakładek — opcji ogólnych i opcji zaawansowanych.

Na pierwszej z wymienionych zakładek znajdują się często zmieniane informacje, czyli np. nazwa strony, treść strony itp.



Natomiast na zakładce opcji zaawansowanych umieścimy rzadziej używane elementy, takie jak znaczniki meta, szablony itp. Zastosujemy nazwę *Zaawansowane* dlatego, iż nazwa *Rzadko używane opcje* byłaby zbyt długa, oraz dlatego, że niektórzy administratorzy mogą nie wiedzieć, do czego służą niektóre z zawartych na tej karcie opcji.

Dodajemy menu z zakładkami do pliku `/ww.admin/pages/forms.php`:

```
// { jeśli strona jest niewidoczna w nawigacji, to ma zostać wyświetlony stosowny komunikat
if($page['special']&2)
    echo '<em>UWAGA: ta strona jest aktualnie niewidoczna w nawigacji. Aby ją
    ↪ przywrócić do widoku, skorzystaj z opcji na karcie Zaawansowane.</em>';
// }
echo '<form id="pages_form" method="post">';
echo '<input type="hidden" name="id" value="',$id,'" />'
    , '<div class="tabs"><ul>'
    , '<li><a href="#tabs-common-details">Opcje ogólne</a></li>'
    , '<li><a href="#tabs-advanced-options">Zaawansowane</a></li>'
    ;
// tu będą zakładki wtyczek
echo '</ul>';
```

Jeśli dana strona nie jest widoczna w menu nawigacyjnym witryny, to nad tym formularzem będzie wyświetlona odpowiednia informacja. Strona nie jest uwzględniana w menu nawigacyjnym wówczas, gdy w polu `special` ma wartość 2.

Maski bitowe są przydatne w sytuacjach, gdy mamy wartości typu „tak” i „nie” i nie chcemy zajmować całego pola w bazie danych na przechowywanie każdej z nich.

Dalej zaczyna się kod generujący formularz.

Zwróć uwagę na brak parametru `action`. Mimo iż w specyfikacji W3C języka HTML 4.01 parametr ten jest wymagany, to żadna przeglądarka nie wymusza jego stosowania. Kiedy go brak, to przeglądarki do przetwarzania danych z formularza stosują domyślnie ten sam plik.

To samo dotyczy elementu `style`, którego typ jest domyślnie ustawiany na `text/css`, i elementu `script`, którego typ jest domyślnie ustawiany na `javascript`.

Za formularzem mamy menu zakładek zbudowane na bazie elementu listy.

W przedostatnim wierszu kodu znajduje się komentarz informujący o zakładkach wtyczek. Kiedy w dalszej części rozdziału zaczniemy zajmować się wtyczkami, niektóre z nich mogą mieć na tyle dużo opcji, że trzeba będzie je wydzielić do osobnej karty. Zajmiemy się tym później.

Teraz dodamy zakładkę opcji ogólnych (pracujemy cały czas na tym samym pliku):

```
// { opcje ogólne
echo '<div id="tabs-common-details"><table
style="clear:right;width:100%;"><tr>';
// { nazwa
echo '<th width="5%">nazwa</th><td width="23%">
  <input
    id="name" name="name"
    value="",htmlspecialchars($page['name']),' /></td>';
// }
// { tytuł
echo '<th width="10%">tytuł</th><td width="23%">
  <input
    name="title"
    value="",htmlspecialchars($page['title']),' /></td>';
// }
// { url
echo '<th colspan="2">';
if($edit){
  $u='/'.str_replace(' ','-',$page['name']);
  echo '<a style="font-weight:bold;color:red" href="',$u,'"
target="_blank">PODGLĄD</a>';
}
else echo '&nbsp;';
echo '</th>';
// }
echo '</tr><tr>';
// { typ
echo '<th>typ</th><td><select name="type"><option value="0">normalna</option>';
// tutaj wstaw typy dodane jako wtyczki
echo '</select></td>';
// }
// { rodzic
```

```

echo '<th>rodzic</th><td><select name="parent">';
if($page['parent']){
    $parent=Page::getInstance($page['parent']);
    echo '<option value="' . $parent->id . '">', htmlspecialchars($parent->name),
        '</option>';
}
else echo '<option value="0"> -- ', 'brak', ' -- </option>';
echo '</select>', "\n\n", '</td>';
// }
if(!isset($page['associated_date']) || !preg_match('/^[0-9]{4}-[0-9]{2}-
↳ [0-9]{2}$/', $page['associated_date']) || $page['associated_date']==
↳ '0000-00-00') $page['associated_date']=@date('Y-m-d');
echo '<th>Data</th><td><input name="associated_date" class="date-human"
↳ value="' . $page['associated_date'] . '" /></td>';
echo '</tr>';
// }
// { dane dotyczące typu strony
echo '<tr><th>treść</th><td colspan="5">';
echo '<textarea name="body">', htmlspecialchars($page['body']), '</textarea>';
echo '</td></tr>';
// }
echo '</table></div>';
// }

```

Ten skrypt wyświetla wartości najczęściej zmienianych pól tabeli bazy danych:

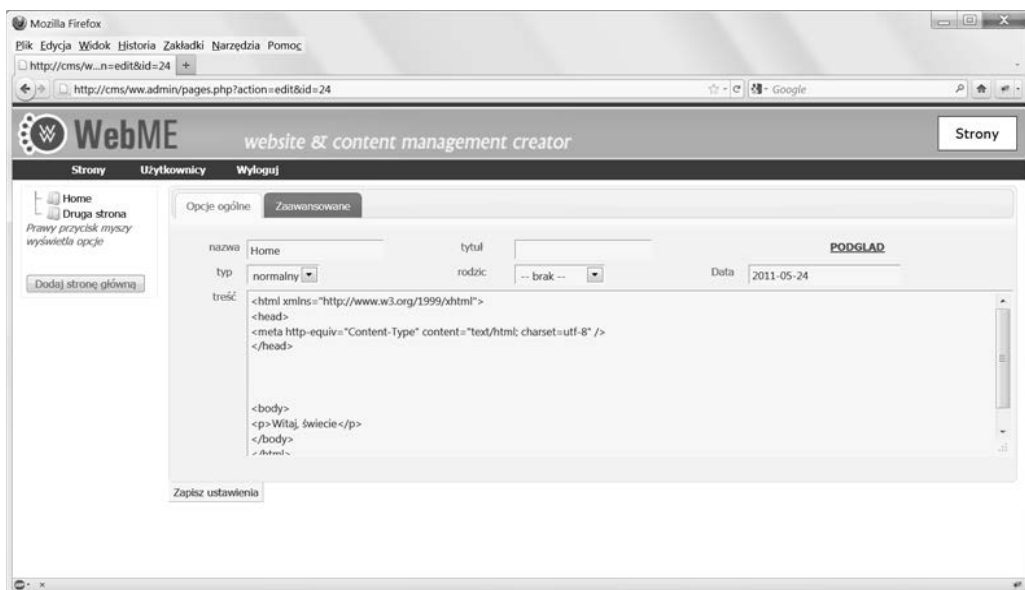
- name (nazwa)
- title (tytuł)
- type (typ)
- parent (rodzic)
- associated_date (data)
- body (treść)

Kilkoma z nich jeszcze się zajmiemy, kiedy skończymy z formularzem. A na razie należy poczynić kilka uwag na temat formularza i zawartych w nim opcji:

- **URL:** podczas modyfikowania strony dobrze jest mieć ją wyświetloną w innym oknie lub na innej karcie. Aby to umożliwić, dodaliśmy odnośnik do strony w widoku przeznaczonym dla użytkowników. Jego kliknięcie powoduje otwarcie nowej karty lub nowego okna.
- **Typ:** domyślnie stronom nadawany jest typ normalny, który jak na razie jest jedyną opcją do wyboru. Później, kiedy dodamy wtyczki, tych możliwości będzie więcej.
- **Rodzic:** to strona, w której znajduje się aktualnie redagowana strona. Wyświetlony jest tylko bieżący rodzic i nie ma żadnych dodatkowych opcji. Jest ważny powód, aby tak było; stanie się on jasny po zakończeniu pracy nad formularzem.

- **Data:** ze stroną związanych jest kilka dat. Wewnętrznie rejestrowane są daty utworzenia i ostatniej modyfikacji (które mogą być przydatne wtyczkom), ale czasami administrator chce określić datę strony. Sytuacja taka może mieć miejsce np. wówczas, gdy strona stanowi część systemu wiadomości. Pole daty jeszcze rozbudujemy, gdy skończymy pracę nad formularzem.
- **Treść:** zawartość, która będzie wyświetlona na froncie strony. W tym polu należy wpisywać zwykły kod HTML. Oczywiście przeciętny administrator niekoniecznie zna język HTML, dlatego trzeba będzie nad tym jeszcze trochę popracować.

Tak teraz wygląda zawartość pierwszej karty (na zrzucie widać już ukończone karty stworzone przy użyciu jQuery — jak to zrobiłem, dowiesz się w dalszej części rozdziału).



Jak widać, pole daty jest dość duże. Nie jest tak bez powodu, o czym przekonasz się w następnym rozdziale.

Kod źródłowy drugiej karty będzie nieco krótszy. Dodaj poniższy kod do pliku `/ww.admin/pages/forms.php`:

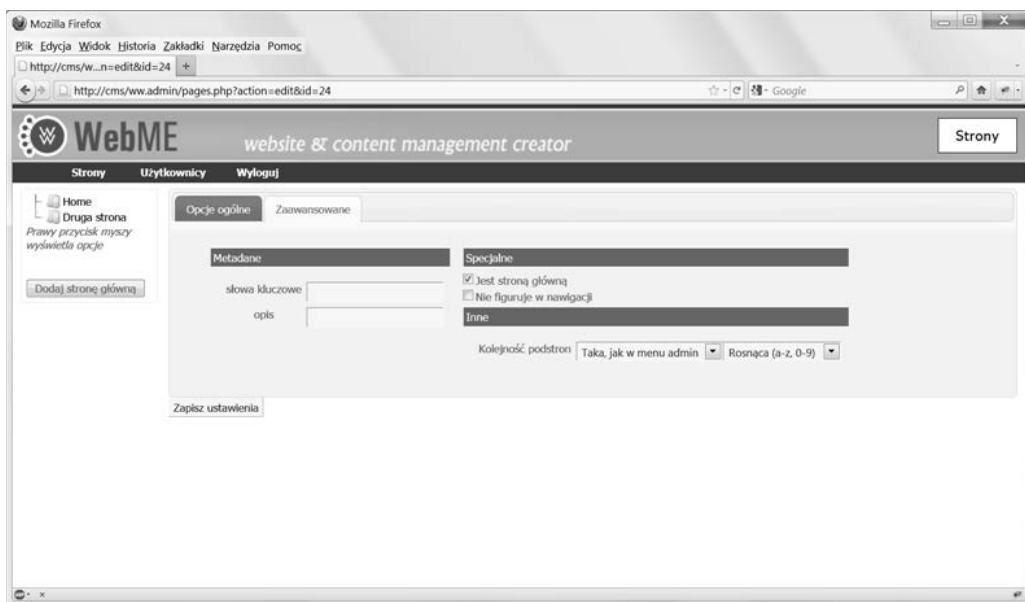
```
// {opcje zaawansowane}
echo '<div id="tabs-advanced-options">';
echo '<table><tr><td>';
// {metadane}
echo '<h4>Metadane</h4><table>';
echo '<tr><th>Słowa kluczowe</th><td>
  <input name="keywords"
    value="" ,htmlspecialchars($page['keywords']), '
  /></td></tr>';
```

```

echo '<tr><th>opis</th><td>
    <input name="description"
        value="" ,htmlspecialchars($page['description']),'
    /></td></tr>';
    // { szablon
// odpowiedni kod zostanie dodany w następnym rozdziale
// }
echo '</table>';
// }
echo '</td><td>';
// { specjalne
echo '<h4>Specjalne</h4>';
$specials=array('Jest stroną główną',
    'Nie figuruje w nawigacji');
for($i=0;$i<count($specials);++$i){
    if($specials[$i]!=''){
        echo '<input type="checkbox" name="special[',$i,']"' ;
        if($page['special']&pow(2,$i))echo ' checked="checked"' ;
        echo ' />',$specials[$i], '<br />';
    }
}
// }
// { inne
echo '<h4>Inne</h4>';
echo '<table>';
// { kolejność podstron
echo '<tr><th>Kolejność podstron</th><td><select name="page_
vars[order_of_sub_pages]">';
$arr=array('Taka, jak w menu admin','Alfabetyczna','by associated date');
foreach($arr as $k=>$v){
    echo '<option value="',$k,'"';
    if(isset($page_vars['order_of_sub_pages']) &&
        $page_vars['order_of_sub_pages']==$k)
        echo ' selected="selected"';
    echo '>',$v,'</option>';
}
echo '</select>';
echo '<select name="page_vars[order_of_sub_pages_dir]">
    <option value="0">Rosnąca (a-z, 0-9)</option>';
echo '<option value="1"' ;
if(isset($page_vars['order_of_sub_pages_dir']) &&
    $page_vars['order_of_sub_pages_dir']=='1')
    echo ' selected="selected"';
echo '>Malejąca (z-a, 9-0)</option></select></td></tr>';
// }
echo '</table>';
// }
echo '</td></tr></table></div>';
// }

```

Nie ma tu wiele do wyjaśniania. Można by było dodać jeszcze kilka zaawansowanych opcji, które byłyby przydatne, gdyby system był bardziej rozwinięty (dodane wtyczki, ukończony system szablonów i motywów itd.).



Najpierw dodane zostały **pole tekstowe** na **słowa kluczowe** i opis (elementy meta keywords i description). Większość ludzi pozostawia je nietknięte i dlatego znalazły się na karcie *Zaawansowane*.

Szablony i motywy zostaną dodane w następnym rozdziale, a na razie w ich zastępstwie w kodzie znajduje się tylko komentarz informujący o przyszłych planach.

Dalej znajduje się lista „specjalnych” cech. Na razie są tylko dwie: pole wyboru pozwalające oznaczyć stronę jako główną oraz pole umożliwiające usunięcie strony z nawigacji.

Na końcu znajdują się dwie listy rozwijane służące do określania kolejności podstron bieżącej strony w nawigacji frontowej. Możesz przykładowo zechcieć posortować alfabetycznie listę autorów albo aby nowe elementy pojawiały się w kolejności malejącej według daty. Najczęściej jednak stosuje się tę samą kolejność, co w menu administracyjnym (można ją zmienić, przeciągając nazwy stron w menu nawigacyjnym po lewej stronie).

Czas na dokończenie formularza i dodanie kodu odpowiedzialnego za wyświetlanie zakładek.

Warto dodać jeszcze jedną sekcję do formularza — niektóre wtyczki mogą wymagać dodawania własnych zakładek, ale do tego wrócimy kiedy indziej.

Dodaj do pliku `/ww.admin/pages/forms.php` poniższy kod:

```
echo '</div><input type="submit" name="action" value="',
      ($edit?'Zapisz ustawienia':'Wprowadź ustawienia')
      ,' " /></form>';
echo '<script>>window.currentpageid='.$id.';</script>';
echo '<script src="/ww.admin/pages/pages.js"></script>';
```

Ponadto utwórz następujący plik `/ww.admin/pages/pages.js`:

```
$(function(){
  $('.tabs').tabs();
});
```

Zmienna `window.currentpageid` będzie potrzebna w następnym podrozdziale.

Podstawowa wersja formularza jest już gotowa.

Teraz zajmiemy się rozszerzaniem funkcjonalności uprzednio wyróżnionych pól.

Asynchroniczne wypełnianie listy rodziców

W bardzo dużych serwisach proces wczytywania formularza może być bardzo powolny z powodu długiej listy rozwijanej *rodzic* informującej serwer, której strony potomkiem jest dana strona.

Jeśli listę tę będzie się wypełniać podczas wczytywania formularza, to ilość kodu HTML do pobrania może być bardzo pokaźna.

Rozwiązanie tego problemu opracowałem na potrzeby mojej poprzedniej książki (*jQuery 1.3 with PHP*). Zaimplementowałem je w postaci wtyczki jQuery, której teraz użyjemy.

Pobierz wtyczkę `remoteselectoptions` ze strony <http://plugins.jquery.com/project/remoteselectoptions> i wypakuj ją do katalogu `/j/`.

Wtyczka ta działa następująco: przy wczytywaniu strony do listy rozwijanej pobierana jest tylko jedna opcja, a pozostałe zostają dobrane dopiero wtedy, gdy są rzeczywiście potrzebne, tzn. gdy zostanie kliknięty element listy.

Aby zmusić ją do takiej współpracy z polem *rodzic*, otwórz plik `/ww.admin/pages/pages.js` i zmień jego zawartość następująco:

```
$(function(){
  $('.tabs').tabs();
  $('#pages_form select[name=parent']).remoteselectoptions({
    url: '/ww.admin/pages/get_parents.php',
    other_GET_params: currentpageid
  });
});
```


Jako że ta wtyczka może być przydatna w wielu miejscach panelu administracyjnego, dodamy ją do pliku */ww.admin/header.php* (wyróżniony fragment):

```
<script src="http://ajax.googleapis.com/ajax/libs
  /jqueryui/1.8.0/jquery-ui.min.js"></script>
<script src="/j/jquery.remoteselectoptions/jquery.remoteselectoptions.js">
  ↪</script>
<link rel="stylesheet" href="http://ajax.googleapis.com/ajax
  /libs/jqueryui/1.8.0/themes/south-street/jquery-ui.css"
  type="text/css" />
```

Z treści pliku *pages.js* wynika, że do budowy listy nazw stron potrzebny jest jeszcze jeden plik — */ww.admin/pages/get_parents.php*. Oto jego zawartość:

```
<?php
require '../admin_libs.php';
function page_show_pagenames($i=0,$n=1,$s=0,$id=0){
  $q=dbAll('select name,id from pages where parent="'. $i.'" and id!="'. $id.'"
  ↪order by ord,name');
  if(count($q)<1)return;
  foreach($q as $r){
    if($r['id']!='){
      echo '<option value="'. $r['id'].'" title=
      ↪"'. htmlspecialchars($r['name']).'";
      echo ($s==$r['id'])? ' selected="selected">': '>';
      for($j=0;$j<$n;$j++)echo '&nbsp;';
      $name=$r['name'];
      if(strlen($name)>20)$name=substr($name,0,17).'...';
      echo htmlspecialchars($name).'</option>';
      page_show_pagenames($r['id'],$n+1,$s,$id);
    }
  }
}
}
}
$selected=isset($_REQUEST['selected'])
  ?$_REQUEST['selected']:0;
$id=isset($_REQUEST['other_GET_params'])
  ?(int)$_REQUEST['other_GET_params']:-1;
echo '<option value="0"> -- brak -- </option>';
page_show_pagenames(0,0,$selected,$id);
```

Wtyczka *remoteselectoptions* wysłała do tej strony żądanie z dwoma parametrami: identyfikatorem aktualnie wybranego rodzica i identyfikatorem aktualnej strony.

Powyższy skrypt tworzy listę opcji oraz uniemożliwia administratorowi wstawienie strony w niej samej ani w stronie, która znajduje się niżej od niej w hierarchii. Gdyby tak się stało, strona zniknęłaby z wszystkich menu nawigacyjnych, także w panelu administracyjnym.

W tej chwili oznacza to, że jedyne dostępne opcje to albo *brak* (jeśli strona jest najwyższego poziomu), albo *Druga strona*, tak jak w naszym przykładzie. Aktualnie są tylko dwie strony i oczywiście nie można umieścić strony *Home* pod stroną *Home*.



Home	tytuł	
normalny	rodzic	-- brak --
<code><html xmlns="http://www.w3.org/1999/xhtml"></code>		

Wykonaliśmy kawał dobrej roboty, więc możesz sobie zrobić przerwę, aby nabrać sił na dokończenie mechanizmu tworzenia stron w następnym rozdziale.

Podsumowanie

W tym rozdziale zostały stworzone podstawy systemu zarządzania stronami. Utworzyliśmy formularz do zarządzania nimi oraz za pomocą narzędzi jQuery zaimplementowaliśmy prosty mechanizm zamiany stron miejscami i optymalizacji wyświetlania długich list opcji.

W następnym rozdziale dokończymy sekcję zarządzania stronami i zbudujemy proste frontowe menu nawigacyjne, dzięki któremu będzie można wchodzić na różne strony.

Skorowidz

A

- administracja okienkami, 242
- administracja stronami, 82
- administrator, 47
- adres ../news, 18
- adres ../news?page=2, 18
- adres 127.0.0.1, 27
- adres e-mail jako nazwa użytkownika, 49
- aktualizacja strony, 99
- algorytm
 - bcrypt, 45
 - SHA1, 45
 - MD5, 45
- analiza składniowa, 25

B

- baza danych dla CMS, 20
- biblioteka
 - ImageMagick, 109
 - PDO, 67
 - reCAPTCHA, 52
 - jQuery UI, 54

C

- CKEditor, 105
- CMS, 19
 - proces działania, 19
 - sekcja prywatna, 20
 - sekcja publiczna, 18

D

- daty, 93
- dodawanie formularzy, 191
- dodawanie wtyczki, 161
- dodawanie zakładek, 178
- dodawanie zdarzeń, 172
- dyrektywa AllowOverride, 29

E

- edycja użytkownika, 69
- edytor FCKeditor, 104
- edytor tekstu sformatowanego, 104
- eksport danych, 212
- element .panel-opener, 259
- element <script>, 54
- element input typu hidden, 101
- encje HTML, 150

F

- Firebug, 81
- format JSON, 44
- formularz
 - Data, 87
 - do konfiguracji widżetów, 259
 - do przypominania hasła, 48
 - do zarządzania stronami, 73
 - form.php, 70
 - logowania, 46, 54
 - Rodzic, 86

formularz
 Treść, 87
 Typ, 86
 URL, 86
 tworzenia stron normalnego typu, 83
 Front CMS, 18
 funkcja
 .tree(), 101
 __autoload(), 112
 buildRightWidget(), 251
 confirm(), 68
 contentsnippet_show(), 256
 dbAll(), 67
 die(), 121
 form_template_generate(), 204
 formfieldsAddRow(), 200
 getRelativeURL(), 129
 htmlspecialchars(), 129
 include(), 25
 is_admin(), 47, 59
 menu_build_fg(), 127, 128
 menu_show_fg(), 126
 onchange(), 80
 onmove(), 81
 outerHeight(), 135
 outerWidth(), 135
 page_comments_admin_page_tab(), 178
 pages_setup_name(), 97
 PANEL, 239
 panel_selectkiddies(), 276
 panels_init(), 245
 panels_show(), 239, 255
 require(), 25
 Save(), 265
 showWidgetForm(), 260
 step2_verify(), 297
 updateWidgets(), 253
 widget_header_visibility(), 269
 widget_rename(), 268
 widget_toggle_disabled(), 269
 funkcja automatycznego ładowania, 36
 funkcja do obsługi przekierowań, 55
 funkcja zapamiętywania zdarzeń, 80
 funkcja zwiżania, 75

G

galeria siatkowa, 232
 generator formularzy, 185
 grupa, 42

H

hierarchia stron, 77, 81

I

identyfikator strony, 35
 instalacja CMS-a w maszynie wirtualnej, 287
 instalacja maszyny wirtualnej, 284
 instalacja narzędzia VMware Player, 284
 integracja menu, 137

J

jądro, 154
 jądro CMS, 17, 290
 język PHP, 116
 jQuery, 15, 50
 jQuery UI, 15, 50

K

katalog
 .private, 25, 27
 f, 109
 j, 78
 kfm, 112
 ww.admin, 23, 46
 ww.admin/pages, 75
 ww.admin/themes, 141
 ww.incs/, 52
 ww.plugins/content-snippet, 246
 ww.plugins/forms, 186
 ww.plugins/forms/admin, 193
 ww.plugins/image-gallery/admin, 217
 trunk, 108
 katalog główny, 24
 katalog kompilacji, 121
 katalog sieciowy, 23
 klucz prywatny (Private Key), 52
 klucz publiczny (Public Key), 52
 klucz do API, 52
 kod frontowy, 278
 kolumna
 activation_key, 44
 active, 44
 associated_date, 33
 body, 32, 239
 cdate, 33
 description, 33

- disabled, 239
- edate, 33
- email, 44
- extras, 44
- groups, 44
- id, 32, 44, 239
- keywords, 33
- name, 32, 239
- ord, 33
- parent, 33
- password, 44
- special, 33
- template, 33
- title, 33
- type, 33
- vars, 33
- visibility, 239
- konfiguracja, 26
- konfiguracja okienka, 247
- konfiguracja wtyczki, 156
- konfiguracja wtyczki galerii obrazów, 216
- konsola MySQL, 26
- kontroler, 19
- kontroler frontu, 29

L

- lista rodzic, 90
- logowanie, 54
- logowanie do narzędzia KFM, 111

M

- magiczne cudzysłowy, 31
- maszyna wirtualna, 284
- mechanizm uaktualnień, 163
- menedżer plików, 107
- menu administracyjne, 74
- menu Filament Group, 134
- menu nawigacyjne, 125
- menu strony, 76
- menu użytkownika, 165
- metoda
 - .destroy(), 265
 - GET, 30
 - getInstance, 38
 - getInstanceByName, 38
 - getInstanceBySpecial, 38
 - POST, 30

- moderacja komentarzy, 170
- moduł mod_deflate, 31
- moduł mod_rewrite, 23, 223
- moduł pobierający dane, 19
- motyw, 116
- motyw Basic, 142
- MVC, Model-View-Controller, 19

N

- narzędzie
 - CKEditor, 104, 105
 - KFM, 107, 110
 - QEMU, 284
 - SuPHP, 109
 - TinyMCE, 105
 - VirtualBox, 284
 - Virtuozzo, 284
 - VMware, 284
 - Xen, 284
- Nolte Tim, 11

O

- obiekt \$PAGEDATA, 158
- obiekt Page, 36
- obsługa serwera, 29
- odnośnik uwierzytelniający, 63
- odwołanie do pliku, 23
- odzyskiwanie hasła, 62
- okienko, 237, 240

P

- panel administracji, 20
- parametr
 - action, 54, 85, 96
 - page=2, 19
 - redirect, 54
- pasek narzędzi, 107
- PHP, 15
- phpMyAdmin, 26
- plik
 - _default.html, 118, 125, 240
 - .htaccess, 24, 28
 - action.php, 69, 71
 - activate.php, 183
 - admin_libs.php, 47
 - admin.js, 95, 279

plik

- basics.php, 34, 67, 112, 143, 159
- CentOS-5.6-i386-bin--DVD.iso, 285
- config.php, 27, 111, 145, 165, 300
- check-config.php, 298
- comments.php, 170
- common.php, 34, 126, 137
- configuration.dist.php, 109
- delete.php, 103, 183
- disable.php, 162
- enable.php, 161
- footer.php, 60
- form.php, 193
- forms.php, 75, 82, 84, 147
- gallery-type-ad.php, 227
- get_parents.php, 91
- get_types.php, 189
- get-visibility.php, 275
- header.php, 60, 91
- httpd.conf, 27
- index.php, 18, 24, 30, 34, 123, 137, 229, 256
- index.html, 30
- jquery.tree.js, 79
- js.js, 259, 271, 273, 277, 292
- list.php, 144, 160
- login.css, 48
- login-codes.php, 57
- login-libs.php, 55
- login.login.js, 50, 54
- login.php, 47, 50, 56
- logout.php, 61
- menu.js, 79
- menu.php, 76, 79, 170
- move_page.php, 82
- news, 18
- page-tab.js, 181
- page-tab.php, 178, 180
- page.php, 34, 36, 149
- pages.action.edit.php, 97
- pages.php, 75, 96
- password-reminder.php, 63
- plugin.php, 156, 169, 186, 201, 238, 246
- recaptcha.php, 52
- remove-panel.php, 273
- screenshot.png, 142
- show.php, 174, 203, 226, 232
- themes.php, 159
- upgrade.php, 163, 187, 239
- uploader.php, 222
- users.php, 66, 159

- plik konfiguracyjny, 25
- plik w formacie wykonywalnym, 26
- podmenu, 139
- podmenu wielokolumnowe, 130
- poła formularza, 197
- pole
 - Eksport, 197
 - email, 197
 - Odpowiedz, 197
- pole bitowe, 98
- proces logowania, 55
- proces wczytywania formularza, 90
- przechwytywanie błędu 404, 39

R

- reguła, 31
- rekursywny obiekt JSON, 166
- resetowanie hasła, 62
- rodzaje użytkowników, 41
- rola, 42
 - _administrators, 43
 - _superadministrators, 43
- rozszerzenie .html, 118

S

- serwis Google Content Delivery Network (CDN), 50
- skórka, 115
- skrypt
 - get.php, 112
 - captcha, 53
 - do tworzenia i edytowania użytkowników, 72
 - Filament Group Menu, 134
 - logowania, 51
 - mod_rewrite, 34
 - obsługi wysyłania formularza, 207
 - uwierzytelniający, 64
 - wylogowania, 61
- słowa kluczowe, 89
- Smarty, 117, 239
 - instalacja, 121
 - konfiguracja, 120, 124
- SQLite, 108
- stała
 - CONFIG_FILE, 36
 - SCRIPTBASE, 36
 - THEME, 120
 - THEME_DIR, 120

strona, 73
 administracyjna, 46
 do zarządzania użytkownikami, 66
 główna, 76
 home, 33
 index.php, 46
 login.php, 54
 logowania, 46
 wyboru motywów, 146
 struktura bazy danych, 24
 struktura katalogów, 22
 struktura plików motywu, 118
 superadministrator, 47
 system logowania, 66
 szablon, 116
 _default, 148
 do prezentacji treści serwisu, 115
 Smarty, 116
 strony, 115

T

tabela
 forms_fields, 188
 forms_saved, 188
 forms_saved_values, 188
 groups, 24
 pages, 24, 32
 user, 24
 user_accounts, 44
 tablica
 \$custom_tabs, 179
 \$DBVARS, 159, 163
 \$gvars, 219
 \$page, 83
 \$plugin, 157, 163
 \$PLUGIN_TRIGGERS, 173
 \$PLUGINS, 160
 admin, 157
 frontend, 187
 menu, 157
 page_tab, 157
 page_type, 187
 page_vars, 83
 triggers, 157
 ww_widgets, 248
 test captcha, 52
 tworzenie instalatora, 290
 tworzenie podstron, 101

tworzenie stron najwyższego poziomu, 99
 tworzenie użytkownika, 69
 tworzenie wtyczki fragmentów treści, 246
 typy stron, 155, 188

U

uprawnienia „0777”, 110
 usługa Akismet, 176
 ustawianie motywów, 140
 ustawienia główne, 195
 usuwanie komentarzy, 183
 usuwanie obrazów, 224
 usuwanie okienek, 273
 usuwanie stron, 102
 usuwanie użytkowników, 68
 uwierzytelnianie użytkownika, 63

V

Verens Kae, 7

W

wczytywanie danych strony, 31, 34
 WebMe, Website Management Engine, 18
 węzeł, 80
 widoczność okienek, 274, 278
 widoczność widżetów, 279
 widżet
 przeciąganie widżetów do okienek, 249
 widoczność nagłówek widżetów, 268
 wyłączenie widżetów, 269
 wyświetlanie widżetów, 248
 zapisywanie zawartości okienka, 252
 zmienianie nazw, 267
 włączanie biblioteki fg-menu, 168
 włączanie komentarzy, 183
 włączanie wtyczki, 158, 161
 wskazanie domeny, 27
 wtyczka, 21, 153
 ad-gallery, 226
 contextmenu, 101
 datepicker, 94
 do tworzenia formularzy, 185
 do tworzenia okienek, 238
 fg-menu, 137
 Fragmenty, 246
 galerii obrazów, 215

wtyczka
 jstree, 78, 101
 Komentarze, 155, 164
 Page Comments, 163
 remoteselectoptions, 90, 91, 189
 wylogowywanie, 60
 wyłączanie komentarzy, 183
 wyłączanie okienek, 271
 wysyłanie wiadomości, 209
 wyświetlanie listy stron, 74
 wyświetlanie okienek, 243
 wyświetlanie strony, 34
 wyświetlenie formularza, 202
 wyzwalacz finish, 155
 wyzwalacz start, 155
 wyzwalacz (trigger), 22

Z

Zabin Paul, 11
 zakładki, 217
 zamykanie menu, 136
 zapisywanie danych, 211
 zapisywanie komentarzy w bazie danych, 176
 zarządzanie użytkownikami, 41

zdarzenie, 22, 155
 page-content-created, 155
 mouseenter, 137
 zmienna
 \$_REQUEST, 56
 \$_SESSION['userdata'], 61
 \$html, 180
 \$htmlurl, 171
 \$kfim_userfiles_address, 109
 \$METADATA, 117
 \$PAGECONTENT, 117
 special, 97
 userdata, 60
 zmienne konfiguracyjne, 25
 znak ?>, 28
 znak _, 43
 znaki (!£\$%^&*?), 127
 znaki {...}, 119

Ż

żądanie HTTP, 19

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>

Projektowanie systemów CMS przy użyciu PHP i jQuery

Na rynku systemów do zarządzania treścią jest ogromny wybór różnorodnych rozwiązań, od Joomla! zaczynając, a na Drupalu i Wordpressie kończąc. Warto jednak zastanowić się, czy nie lepiej stworzyć własne, skrojone na miarę rozwiązanie, idealnie dopasowane do potrzeb i możliwości. Zamiast przedzierać się przez zawiłą konfigurację, przygotowywać szablony i próbować pogodzić ze sobą wtyczki, być może lepiej tę energię spożytkować na napisanie własnego CMS-a?

Jeżeli zdecydujesz się na to, w niniejszej książce znajdziesz szczegółowy przewodnik, jak dokonać tego przy użyciu najpopularniejszych narzędzi: łącząc język PHP i bibliotekę jQuery. W trakcie lektury dowiesz się, jak zaprojektować jądro swojego nowego systemu, moduł zarządzania użytkownikami oraz własny mechanizm szablonów. Ponadto nauczysz się zarządzać treścią, tworzyć hierarchię stron oraz edytować je przy użyciu wygodnych narzędzi, takich jak CKEditor. Na koniec zbudujesz instalator swojego CMS-a oraz wzbogacisz go o obsługę wtyczek i widżetów. Książka ta jest pozycją obowiązkową dla każdego webmastera pragnącego stworzyć własny, unikatowy system zarządzania treścią.

Zbuduj swój autorski system CMS, wykorzystując niezastąpiony duet PHP i jQuery!

Główne zagadnienia omówione w książce:

- jądro systemu CMS,
- panel administracyjny,
- struktura bazy danych oraz katalogów,
- zarządzanie użytkownikami – role, rodzaje użytkowników,
- logowanie użytkowników,
- procedura odzyskiwania hasła,
- wyświetlanie listy stron,
- tworzenie, edycja i usuwanie stron,
- wykorzystanie narzędzia CKEditor,
- tworzenie i obsługa szablonów,
- rozszerzanie funkcjonalności za pomocą wtyczek,
- budowa instalatora.

helion.pl
księgarnia internetowa

Nr katalogowy: **7102**

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. Kosciuszki 1c, 44-100 Gliwice

tel.: 32 230 98 83

e-mail: helion@helion.pl

<http://helion.pl>

sięgnij po **WIĘCEJ**



KOD KORZYŚCI

ISBN 978-83-246-3365-4



Cena 59,00 zł

Informatyka w najlepszym wydaniu