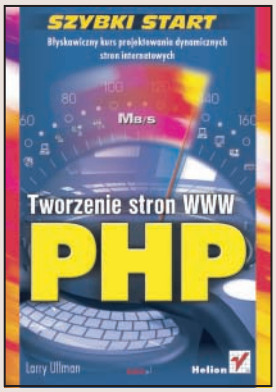
PHP. Tworzenie stron WWW. Szybki start



Autor: Larry Ullman
Tłumaczenie: Paweł Gonera
ISBN: 83-7361-530-X
Tytuł oryginału: PHP for the World Wide Web Visual QuickStart Guide, 2nd Edition
Format: B5, stron: 456

Błyskawiczny kurs tworzenia dynamicznych stron WWW z wykorzystaniem języka PHP

Język PHP cieszy się zasłużoną popularnością wśród twórców stron WWW. Z mało znanego produktu open source szybko stał się cenionym narzędziem stosowanym do tworzenia dynamicznych witryn sieciowych opartych na bazach danych. PHP jest wykorzystywany w portalach, sklepach internetowych, forach dyskusyjnych i wielu innych witrynach z dynamicznie generowaną zawartością. Jeśli język HTML i pisane w nim statyczne strony HTML nie odpowiadają już Twoim wymaganiom, zrób następny krok — poznaj język PHP.

„PHP. Tworzenie stron WWW. Szybki start" to idealny podręcznik dla Ciebie. Dowiesz się z niego wszystkiego, co jest niezbędne do wykorzystywania języka PHP w pracy webmastera. W prostych ćwiczeniach poznasz podstawy języka PHP i nauczysz się stosować go do różnych zadań związanych z tworzeniem stron WWW. Wiadomości zawarte w książce dotyczą zarówno wersji 4, jak i wchodzącej właśnie na rynek wersji 5.

- Podstawy składni języka PHP
- Osadzanie kodu PHP w kodzie HTML
- Zmienne i ich typy
- Obsługa formularzy HTML
- Instrukcje sterujące działaniem programu
- Tablice
- Obsługa sesji i plików cookie
- Praca z systemem plików
- Korzystanie z baz danych
- Stosowanie wyrażeń regularnych

Wydawnictwo Helion
ul. Chopina 6
44-100 Gliwice
tel. (32)230-98-63
e-mail: helion@helion.pl

Spis treści

Formularze HTML i PHP 3

W poprzednim rozdziale przedstawiliśmy krótkie wprowadzenie do zmiennych. Choć bardzo często będziemy tworzyli własne zmienne, to jednak najczęściej będziemy je wykorzystywali w połączeniu z formularzami HTML. Formularze, jak na pewno się już przekonałeś, są podstawowym mechanizmem we współczesnych stronach WWW, pozwalającym zrealizować takie funkcje, jak: rejestracja, logowanie, przeszukiwanie oraz obsługa sklepów internetowych. Korzyści płynące z używania formularzy widać już w najprostszych witrynach WWW. Łącząc je z kodem PHP, możemy niezwykle łatwo odczytywać i obsługiwać generowane przez nie dane.

W rozdziale tym opiszemy podstawy tworzenia formularzy HTML oraz sposoby przesyłania danych do skryptu PHP. Jednocześnie rozdział wprowadza kilka kluczowych mechanizmów prawdziwego programowania w PHP — w tym sposoby uruchamiania skryptów oraz obsługę błędów.

Tworzenie prostego formularza

Jako przykład formularza HTML utworzymy tutaj stronę informacji zwrotnej, która pobiera powitanie użytkownika, jego imię, adres e-mail, odpowiedź oraz komentarz. Będziemy musieli utworzyć odrębne pola dla wszystkich wymienionych tutaj danych. Na początek jednak wrócimy do tematu XHTML.

Jak wspomnieliśmy w pierwszym rozdziale (w którym poinformowaliśmy, że w książce tej będziemy korzystać z XHTML), język ten ma kilka zasad, które powodują, że ma on wyraźnie inną składnię niż HTML. Przypominamy, że kod musi być zapisany małymi literami, a każdy atrybut znacznika należy ująć w cudzysłowy. Dodatkowo, każdy znacznik musi być zamknięty; te, które nie mają formalnego znacznika zamykającego, jak na przykład `<input>`, są zamykane przez dodanie spacji i ukośnika na jego końcu. Dlatego w HTML można napisać:

```
<INPUT TYPE=TEXT NAME=adres SIZE=40>
```

Ale w XHTML to samo wyrażenie musi przyjąć postać:

```
<input type="text" name="adres" size="40" />
```

Powyższe wyjaśnienie pozwoli uniknąć pomyłek w przypadku naszej strony wykorzystującej standard XHTML.

Aby rozpocząć tworzenie podstawowego formularza HTML:

1. Otwórz edytor tekstu i rozpocznij nowy dokument (listing 3.1):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
➥1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/
        ➥xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="content-type"
    ➥content="text/html;
    ➥charset=iso-8859-2" />
    <title>Formularz z opinią</title>
</head>
```

Listing 3.1. *Każdy formularz HTML rozpoczyna się i kończy znacznikami <form></form>. Gdy kodujesz formularze ręcznie — uważaj, aby nie zapomnieć żadnego z tych znaczników. Pamiętaj również o przekierowaniu formularza do właściwego skryptu obsługi za pomocą atrybutu action*

Listing

```
 1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
   1.0 Transitional//EN"
 2         "http://www.w3.org/TR/xhtml1/DTD/
           xhtml1-transitional.dtd">
 3 <html xmlns="http://www.w3.org/1999/xhtml">
 4 <head>
 5     <meta http-equiv="content-type"
       content="text/html;
       charset=iso-8859-2" />
 6     <title>Formularz z opinią</title>
 7 </head>
 8 <body>
 9 Proszę wypełnić ten formularz, aby
   przekazać nam swoje uwagi: <br />
10
11 <form action="obsluga_formularz.php"
   method="post">
12
13 </form>
14
15 <!-- Listing 3.1 - opinia.html -->
16 </body>
17 </html>
```

```
<body>
Proszę wypełnić ten formularz, aby
➥przekazać nam swoje uwagi: <br />
```

2. Dodaj otwierający i zamykający znacznik `<form>`:

```
<form action="obsluga_formularz.php">
</form>
```

Znaczniki `<form>` określają początek i koniec formularza. Każdy element tego formularza musi być zdefiniowany pomiędzy tymi dwoma wierszami. Atrybut `action` wskazuje serwerowi stronę (lub skrypt), która otrzyma dane wprowadzone do formularza, dlatego jest on jednym z najważniejszych elementów.

3. Zakończ stronę, dodając znaczniki zamykające dokument.

```
<!-- Listing 3.1 - opinia.html -->
</body>
</html>
```

4. Zapisz stronę pod nazwą `opinia.html`.

Wskazówki

- Zwróć uwagę, że użyliśmy tutaj rozszerzenia dokumentu właściwego dla HTML (*.html*), ponieważ jest to standardowa strona HTML (a nie strona PHP). Bez problemu można zastosować rozszerzenie *.php*, nawet pomimo to, że nie ma w tym dokumencie kodu PHP (pamiętaj, że na stronie PHP wszystkie elementy poza znacznikami `<?php` i `?>` są traktowane jako kod HTML).
- Upewnij się, że atrybut `action` prawidłowo wskazuje na istniejący na serwerze plik, ponieważ w przeciwnym wypadku formularz nie zostanie właściwie przetworzony. W naszym przykładzie określiliśmy, że formularz ma być przesłany do strony *obsluga_formularz.php*, która jest umieszczona w tym samym katalogu, co strona *opinia.html*.

Po tym omówieniu podstawowych znaczników formularzy dodajmy do niego różne elementy (pola tekstowe przeznaczone na nazwę użytkownika, adres e-mail itd.).

Aby utworzyć elementy formularza:

1. Po wierszu z otwierającym znacznikiem `form` (wiersz 11.), ale przed zamykającym znacznikiem `form` naciśnij *Enter*, aby utworzyć nowy wiersz.

2. Dodaj dwa przyciski opcji przeznaczone na tytuł osoby (listing 3.2):

```
Pan <input type="radio" name="tytul"
➥value="Pan" />
Pani <input type="radio" name="tytul"
➥value="Pani" />
<br />
```

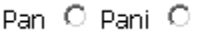
   Ten kod HTML powoduje utworzenie dwóch przycisków opcji (w postaci wybieranych kółeczek, rysunek 3.1). Ponieważ oba przyciski posiadają taką samą wartość atrybutu `name`, może być zaznaczony tylko jeden z nich. Zgodnie z zasadami XHTML, kod jest zapisywany małymi literami (poza wartościami), a na końcu każdego znacznika `input` dodana jest spacja oraz ukośnik domykający znacznik.

   Aby uzyskać odpowiednie rozmieszczenie formularza, na końcu dodaliśmy znacznik `<br />`.

3. Dodaj pola tekstowe przeznaczone na nazwę użytkownika oraz adres e-mail:

```
Imię: <input type="text" name="imie"
➥size="20" />
<br />
Adres e-mail: <input type="text"
➥name="email" size="20" />
<br />
```

   Należy stosować spójną konwencję nazewnictwa elementów formularza i nadawać każdemu elementowi logiczną i opisową nazwę. Przy nadawaniu nazw polom formularza można stosować litery, cyfry oraz znaki podkreślenia (_). Zamieszczone tutaj pola `text` powodują utworzenie w przeglądarce pól tekstowych (rysunek 3.2).



Rysunek 3.1. *Przyciski opcji tytułu*

Rysunek 3.2. *Znaczniki text powodują utworzenie takich pól tekstowych*

Listing 3.2. *Do formularza można dodać dowolną kombinację elementów wprowadzania danych — należy się tylko upewnić, że znajdują się one wewnątrz znaczników <form>, ponieważ inaczej nie pokażą się na stronie*

Listing

```
 1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 2         "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
 3 <html xmlns="http://www.w3.org/1999/xhtml">
 4 <head>
 5     <meta http-equiv="content-type" content="text/html; charset=iso-8859-2" />
 6     <title>Formularz z opinią</title>
 7 </head>
 8 <body>
 9 Proszę wypełnić ten formularz, aby przekazać nam swoje uwagi: <br />
10
11 <form action="obsluga_formularz.php">
12
13 Pan <input type="radio" name="tytul" value="Pan" />
14 Pani <input type="radio" name="tytul" value="Pani" />
15 <br />
16 Imię: <input type="text" name="imie" size="20" />
17 <br />
18 Adres e-mail: <input type="text" name="email" size="20" />
19 <br />
20 Opinia: <select name="opinia">
21 <option value="świetnie">To jest świetne.</option>
22 <option value="w porządku">To jest w porządku.</option>
23 <option value="nuda">To jest nudne.</option>
24 </select>
25 <br />
26 Komentarz: <textarea name="komentarz" rows="3" cols="30"></textarea>
27 <br />
28 <input type="submit" name="wyslij" value="Wyślij opinię" />
29
30 </form>
31
32 <!-- Listing 3.2 - opinia.html -->
33 </body>
34 </html>
```

4. Dodaj listę wyboru opinii:

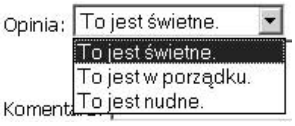
```
Opinia: <select name="opinia">
<option value="świetnie">To jest
➥świetne.</option>
<option value="w porządku">To jest
➥w porządku.</option>
<option value="nuda">To jest nudne.
➥</option>
</select>
<br />
```

Znaczniki te powodują dodanie rozwijanej listy wyboru, jednego z najpopularniejszych typów pól formularza (rysunek 3.3). Każda z opcji umieszczonych pomiędzy znacznikami `select` może zostać wybrana przez użytkownika.

5. Dodaj wielowierszowe pole tekstowe, w którym można napisać własny komentarz:

```
Komentarz: <textarea name="komentarz"
➥rows="3" cols="30"></textarea>
<br />
```

Wielowierszowe pole tekstowe (rysunek 3.4) udostępnia użytkownikowi więcej miejsca na komentarz niż zwykłe pole tekstowe. Jednak pole tekstowe pozwala na ograniczenie liczby znaków, które może wprowadzić użytkownik, natomiast nie da się tego zrobić w przypadku wielowierszowego pola tekstowego (bez użycia dodatkowego kodu JavaScript). W czasie tworzenia formularza należy wybierać typ pola tekstowego najbardziej odpowiedni dla danych, jakie będziesz chciał uzyskać od użytkownika.



Rysunek 3.3. *Znacznik select pozwala utworzyć listę rozwijaną*

Rysunek 3.4. *Znacznik textarea pozwala utworzyć duże pole tekstowe*

Rysunek 3.5. *Każdy formularz musi mieć przycisk wysyłania danych, który może wyglądać inaczej w różnych przeglądarkach*

Rysunek 3.6. *Jeżeli prawidłowo wprowadziłeś kod formularza, powinien on wyglądać w przeglądarce w następujący sposób*

6. Dodaj przycisk wysyłania danych:

```
<input type="submit" name="wyslij"
➥value="Wyślij opinię" />
```

Wartość atrybutu `value` w elemencie typu `submit` jest wyświetlana na przycisku w przeglądarce WWW (rysunek 3.5). Można również tutaj zastosować na przykład słowo *Dalej*.

7. Zapisz skrypt i prześlij go na serwer (lub zapisz w odpowiednim katalogu swojego komputera z zainstalowanym kodem PHP) i wyświetl go w przeglądarce (rysunek 3.6). Ponieważ jest to strona HTML, a nie skrypt PHP, możesz obejrzeć go w przeglądarce bezpośrednio z Twojego komputera.

Wskazówka

- W tym przykładzie utworzyłeś formularz przez ręczne kodowanie w HTML, ale jeżeli wolisz, to samo można zrobić korzystając z aplikacji do projektowania stron WWW (na przykład w programach Macromedia Dreamweaver lub Adobe GoLive).

Użycie GET oraz POST

Doświadczeni autorzy stron HTML zauważą, że w formularzu opinii brakuje jednego elementu: otwierający znacznik `<form>` nie ma atrybutu `method`. Atrybut ten informuje serwer, w jaki sposób przesłać dane do skryptu obsługi.

Atrybut `method` może przyjmować dwie wartości: `GET` oraz `POST`. Wielu koderów HTML nie do końca zna różnice między tymi wartościami i ma wątpliwości, kiedy ich używać. W rzeczywistości dla większości programistów nie ma to najmniejszego znaczenia (szczególnie, gdy zaczynamy dopiero z nich korzystać), ponieważ obie metody pozwalają zazwyczaj na osiągnięcie oczekiwanych wyników.

Różnica między metodami `GET` oraz `POST` polega na sposobie przesyłania danych z formularza do obsługującego go skryptu. Metoda `GET` przesyła wszystkie zebrane informacje jako część adresu URL. Metoda `POST` przesyła dane w sposób niewidoczny dla użytkownika. Na przykład, jeżeli użyjesz metody `GET` w czasie przesyłania danych formularza, wynikowy adres URL będzie miał następującą postać:

```
http://www.serwer.com/phpppr/obsluga_formularz
➥.php?tytul=Pan&nazwa=Larry%20Ullman ...
```

Tymczasem jeśli zastosujemy metodę `POST`, końcowy użytkownik zobaczy adres w postaci skróconej:

```
http://www.serwer.com/phpppr/
➥obsluga_formularz.php
```

Wybierając jedną z tych metod, należy pamiętać o następujących trzech zasadach:

- W przypadku metody `GET` można przesłać tylko ograniczoną ilość informacji.
- Metoda `GET` wysyła dane do skryptu obsługi w sposób jawny. Przykładowo oznacza to, że hasło wprowadzone w formularzu może być przeczytane przez każdego, kto patrzy na przeglądarkę. Warto zauważyć, że jest to dużym zagrożeniem dla bezpieczeństwa.
- Strona wygenerowana przez formularz korzystający z metody `GET` może zostać zapisana w zakładkach, natomiast nie jest to możliwe, gdy dane są przesyłane metodą `POST`.

Listing 3.3. *Aby dokończyć nasz formularz, dodajemy atrybut method z wartością post*

Listing

```
 1 <!DOCTYPE html PUBLIC "-//W3C//DTD
   XHTML 1.0 Transitional//EN"
 2         "http://www.w3.org/TR/xhtml1/DTD/
           xhtml1-transitional.dtd">
 3 <html xmlns="http://www.w3.org/1999/xhtml">
 4 <head>
 5     <meta http-equiv="content-type"
       content="text/html; charset=
       iso-8859-2" />
 6     <title>Formularz z opinią</title>
 7 </head>
 8 <body>
 9 Proszę wypełnić ten formularz, aby
   przekazać nam swoje uwagi: <br />
10
11 <form action="obsluga_formularz.php"
   method="post">
12
13 Pan <input type="radio" name="tytul"
   value="Pan" />
14 Pani <input type="radio" name="tytul"
   value="Pani" />
15 <br />
16 Imię: <input type="text" name="imie"
   size="20" />
17 <br />
18 Adres e-mail: <input type="text"
   name="email" size="20" />
19 <br />
20 Opinia: <select name="opinia">
21 <option value="świetnie">To jest świetne.
   </option>
22 <option value="w porządku">To jest
   w porządku.</option>
23 <option value="nuda">To jest nudne.
   </option>
24 </select>
25 <br />
26 Komentarz: <textarea name="komentarz"
   rows="3" cols="30"></textarea>
27 <br />
28 <input type="submit" name="wyslij"
   value="Wyślij opinię" />
29
30 </form>
31
32 <!-- Listing 3.3 - opinia.html -->
33 </body>
34 </html>
```

Rysunek 3.7. *Przejrzyj źródło strony HTML, aby sprawdzić jej poprawność*

W tej książce do obsługi formularzy niemal wyłącznie stosujemy metodę `POST`, choć przedstawimy przydatną technikę, gdzie metoda `GET` pozwala zyskać dodatkowe możliwości dla stron WWW (patrz podrozdział „Ręczne wysyłanie danych do strony" pod koniec bieżącego rozdziału). Dzięki każdej z tych metod można przesyłać dane z formularza, a ostateczny wybór metody powinien zależeć od potrzebnego Ci poziomu zabezpieczeń oraz od tego, czy wynikowa strona powinna być zapisywana w zakładkach. Nabyte doświadczenie pozwoli szybko rozróżniać obydwie sytuacje, a na razie bezpiecznie możesz w większości przypadków korzystać z metody `POST`.

Aby dodać znacznik method do skryptu:

1. Otwórz plik *opinia.html* w edytorze tekstu (listing 3.2).
2. Wewnątrz otwierającego znacznika `<form>` dodaj `method="post"` (listing 3.3, wiersz 11.). Atrybut formularza `method` wskazuje przeglądarce, w jaki sposób ma ona wysłać dane do skryptu odbierającego.
3. Zapisz skrypt i obejrzyj go ponownie w twojej przeglądarce WWW.
4. Obejrzyj źródło strony, aby upewnić się, że wszystkie potrzebne elementy znajdują się na swoim miejscu. (rysunek 3.7).

Wskazówka

- W opisie metod `GET` oraz `POST` przy ich zapisywaniu korzystaliśmy z wielkich liter. Jednak w formularzu dla zachowania zgodności z XHTML korzystamy z małych liter (`post`). Nie należy przejmować się niespójnością — metoda będzie funkcjonowała prawidłowo niezależnie od wielkości liter.

Odbieranie danych z formularza w PHP

Po utworzeniu prostego formularza HTML musimy napisać skrypt pod nazwą *obsluga_formularz.php*, który otrzymuje i przetwarza dane wygenerowane przez stronę *opinia.html*. W tym konkretnym przykładzie skrypt PHP tylko powtarza wartości wprowadzone przez użytkownika do formularza. W kolejnych rozdziałach pokażemy, w jaki sposób można zapisać dane w bazie danych lub wysłać je korzystając z poczty elektronicznej.

Dla wielu Czytelników przykład ten będzie zaskakująco prosty i utworzony skrypt będzie działał bez zarzutu. Niektórzy mogą jednak napotkać niewielkie problemy spowodowane znacznymi zmianami w domyślnej konfiguracji PHP. W wielu instalacjach PHP można odwoływać się do wartości z formularza poprzez zmienną o nazwie takiej samej, jak nazwa elementu formularza. W omawianym przykładzie zmiennymi tymi są: `$tytul`, `$email`, `$opinia`, `$komentarz` oraz `$wyslij`.

Napisaliśmy powyższy skrypt korzystając ze standardowej metody, która powinna działać u większości Czytelników. Jeżeli tak nie będzie, to w następnym podrozdziale odnajdziesz opisy alternatywnych technik uruchamiania programów. Jeżeli więc w przypadku tego przykładu nie będziesz mógł osiągnąć oczekiwanych wyników, nie musisz się martwić. Następne trzy podrozdziały wszystko wyjaśnią i przy okazji nauczą kilku wartościowych metod.

Aby utworzyć skrypt PHP:

1. Otwórz edytor tekstu i utwórz w nim nowy dokument (listing 3.4):

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
➥1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/
        ➥xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="content-type"
    ➥content="text/html;
    ➥charset=iso-8859-2" />
```

Listing 3.4. *Biorąc nazwę elementu z formularza HTML oraz dodając znak dolara ($), tworzymy zmienne zawierające wartości wprowadzone przez użytkownika w odpowiednim polu*

Listing

```
 1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
   1.0 Transitional//EN"
 2
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
 3 <html xmlns="http://www.w3.org/1999/xhtml">
 4 <head>
 5     <meta http-equiv="content-type"
       content="text/html;
       charset=iso-8859-2" />
 6     <title>Twoja opinia</title>
 7 </head>
 8 <body>
 9 <?php // Listing 3.4 obsluga_formularz.php
10
11 // Strona ta otrzymuje dane z opinia.html.
12 // Otrzymuje: tytuł, nazwę, e-mail,
   opinię oraz komentarz.
13
14 print "$tytul $imie, dziękujemy za
   komentarz. <br />";
15 print "Przykład ten został uznany za
   $opinia i dopisano: $komentarz";
16 ?>
17 </body>
18 </html>
```

```
<title>Twoja opinia</title>
</head>
<body>
```

2. Wstaw otwierający znacznik PHP i wpisz komentarz:

```
<?php // Listing 3.4 obsluga_formularz.php
// Strona ta otrzymuje dane z opinia.html.
// Otrzymuje: tytuł, nazwę, e-mail, opinię
➥oraz komentarz.
```

Komentarze do skryptu dodajemy, aby pomagały zapamiętać, jakie jest jego przeznaczenie. Choć w kodzie strony *opinia.html* zapisane jest, gdzie są wysyłane dane (poprzez atrybut `action`), ten komentarz wskazuje zależność odwrotną, tzn. z którego skryptu otrzymujemy dane.

3. Wyświetl informacje o użytkowniku:

```
print "$tytul $imie, dziękujemy za
➥komentarz. <br />";
print "Przykład ten został uznany za
➥$opinia i dopisano: $komentarz";
```

Aby skorzystać z danych, które użytkownik wpisał do pola `nazwa`, odwołujemy się do zmiennej `$nazwa`. To samo dotyczy wszystkich elementów formularza, niezależnie od ich typu.

4. Zamknij sekcję PHP i zakończ stronę HTML.

```
?>
</body>
</html>
```

5. Zapisz skrypt pod nazwą `obsluga_formularz.php`.

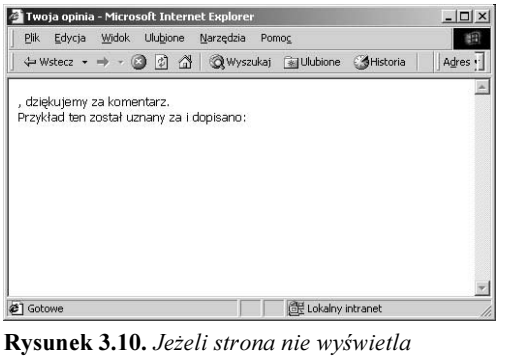
6. Prześlij skrypt na serwer (lub zapisz w odpowiednim katalogu swojego komputera z zainstalowanym PHP), pamiętając, że trzeba go umieścić w tym samym katalogu, co *opinia.html*.

7. Sprawdź skrypt w przeglądarce WWW otwierając stronę *opinia.html*, a następnie wysyłając dane formularza (rysunki 3.8 oraz 3.9).

Jeżeli wartości zmiennych nie są wyświetlane, gdy uruchomisz skrypt albo nie są wyświetlane i widzisz wiele komunikatów błędów, jest to związane ze wspomnianymi wcześniej kłopotami. Niektórzy mogą nic nie zobaczyć (pokaże się pusta strona). Dla każdego z tych przypadków Czytelnik znajdzie rozwiązanie w trzech następnych podrozdziałach.

Rysunek 3.8. *Wszystko, co użytkownik wpisze do formularza HTML, powinno być wyświetlone w przeglądarce WWW przez skrypt obsluga_formularz.php (rysunek 3.9)*

Rysunek 3.9. *Jest to inne zastosowanie instrukcji print omówionej w rozdziale 1., ale dzięki niemu powstała nasza pierwsza wygenerowana dynamicznie strona WWW*



Rysunek 3.10. *Jeżeli strona nie wyświetla przesłanych informacji, przeczytaj podrozdział „Problem z register_globals”*

Wskazówki

- Jeżeli chcesz przekazać do skryptu ustaloną wcześniej wartość, to możesz skorzystać z ukrytego elementu formularza. Na przykład, jeżeli do formularza wstawimy wiersz:

```
<input type="hidden" name="ta_strona"
➥value="opinia.html" />
```

  spowoduje to utworzenie w skrypcie przetwarzającym zmiennej `$ta_strona` posiadającej wartość *opinia.html*.

- Zwróć uwagę, że wartości przycisków opcji oraz wybrane pozycje z listy rozwijanej mają wartość atrybutu `value` wybranego elementu (na przykład *świetnie* w liście wyboru). Taka sama zasada obowiązuje dla pól wyboru. W przypadku pól tekstowych wartością zmiennej jest tekst wpisany przez użytkownika.
- Nawet przycisk przesyłania jest wysyłany do strony obsługi w postaci zmiennej.
- Jeżeli skrypt *obsluga_formularz.php* w wysłanych ciągach wyświetla dodatkowe ukośniki, zapoznaj się z podrozdziałem „Panowanie nad mechanizmem `magic quote`” z rozdziału 5. — „Użycie ciągów”.
- Powinieneś wiedzieć, że przedstawiona tutaj metoda obsługi danych formularza (odwoływanie się do `$nazwa`, `$tytul` itd.) nie jest zalecaną metodą i może powodować naruszenie systemu bezpieczeństwa. Ale pomimo to, oczywiście działa ona skutecznie. Wkrótce przedstawimy lepszą i bardziej bezpieczną metodę, ale na początek chcieliśmy pokazać możliwie proste rozwiązanie.

Wyświetlanie błędów

Po zainstalowaniu PHP na serwerze WWW będzie on działał zgodnie z domyślnymi ustawieniami zabezpieczeń, sposobu obsługi danych, wydajności itd. Od czasu pierwszego wydania tej książki do domyślnej konfiguracji zostały wprowadzone trzy znaczące zmiany i wszyscy programiści PHP powinni o tym wiedzieć. Pierwsza zmiana dotyczy tego, kiedy są wyświetlane komunikaty błędów.

PHP może być ustawiony w taki sposób, że będzie informował nas w przypadku napotkania problemów. To, czy faktycznie to zrobi, zależy od ustawienia parametru `display_errors` w pliku konfiguracyjnym PHP (o nazwie *php.ini*). W bieżącej wersji PHP parametr ten jest domyślnie wyłączony, więc błędy generowane przez skrypt powodują wyświetlenie pustej strony. Mogłeś zobaczyć wynik takiego działania w przypadku poprzedniego skryptu.

Jeżeli chcesz, żeby PHP wyświetlał błędy, to powinieneś wykonać następujące operacje:

- Włącz z powrotem parametr `display_errors` (więcej informacji znajdziesz w podrozdziale „Konfigurowanie PHP” w dodatku A — „Instalacja i konfiguracja”).
- Włącz parametr `display_errors` w samym skrypcie.

Chociaż pierwsza możliwość jest najlepsza dla osób uczących się dopiero PHP, nadaje się tylko dla tych, którzy mogą administrować serwerem. Druga opcja może być użyta przez każdego, wystarczy dodać do skryptu wiersz:

```
ini_set('display_errors', 1);
```

Funkcja `ini_set()` pozwala na tymczasową zmianę w skrypcie ustawień pliku konfiguracyjnego PHP. W tym przykładzie ustawiamy parametr `display_errors` na *on*, co reprezentuje liczba 1.

Aby wyświetlać błędy:

1. Otwórz w edytorze plik *obsluga_formularz.php*.
2. W pierwszym wierszu kodu PHP (listing 3.5) wprowadź następujący kod:

   ```
   ini_set('display_errors', 1);
   ```

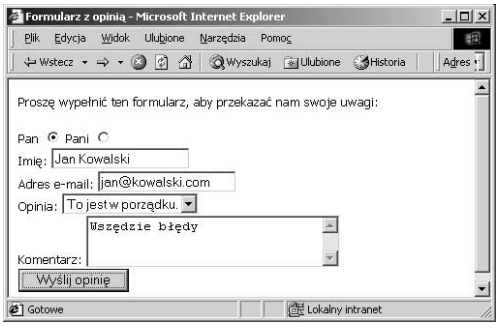
   Powyższy wiersz pozwoli zobaczyć komunikaty o występujących błędach. Powinieneś wpisać ten wiersz jako pierwszą instrukcję PHP, dzięki czemu pozostała część kodu PHP będzie wykonywana z nowymi ustawieniami konfiguracji.
3. Zapisz plik pod nazwą `obsluga_formularz.php`.
4. Wyślij plik na serwer WWW i sprawdź go w przeglądarce (rysunki 3.11 oraz 3.12).

   Jeżeli wynikowa strona nie posiada błędów, skrypt będzie działał tak, jak do tej pory. Jeżeli jednak poprzednio widziałeś pustą stronę, teraz powinieneś zobaczyć komunikat błędu, tak jak na rysunku 3.12. Gdy tak się stanie, rozwiązanie problemu znajdziesz w dalszej części tego rozdziału.

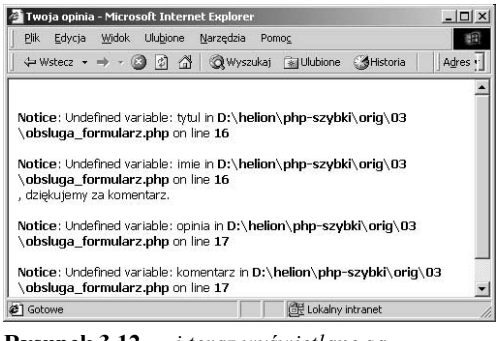
Listing 3.5. *Włączenie dyrektywy display_errors powinno wyeliminować wyświetlanie pustej strony w przypadku wystąpienia błędu*

Listing

```
 1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
   1.0 Transitional//EN"
 2         "http://www.w3.org/TR/xhtml1/DTD/
           xhtml1-transitional.dtd">
 3 <html xmlns="http://www.w3.org/1999/xhtml">
 4 <head>
 5     <meta http-equiv="content-type"
       content="text/html;
       charset=iso-8859-2" />
 6     <title>Twoja opinia</title>
 7 </head>
 8 <body>
 9 <?php // Listing 3.5 obsluga_formularz.php
   (wersja druga)
10
11 ini_set ('display_errors', 1);
   // Pozwól mi uczyć się na błędach.
12
13 // Strona ta otrzymuje dane z opinia.html.
14 // Otrzymuje: tytuł, nazwę, e-mail,
   opinię oraz komentarz.
15
16 print "$tytul $imie, dziękujemy za
   komentarz. <br />";
17 print "Przykład ten został uznany za
   $opinia i dopisano: $komentarz";
18
19 ?>
20 </body>
21 </html>
```



Rysunek 3.11. *Jeszcze raz wypełniamy formularz…*



Rysunek 3.12. *…i teraz wyświetlane są komunikaty o błędach*

Wskazówki

- Jeżeli masz problemy z uruchamianiem skryptu, pamiętaj o włączeniu opcji `display_errors`. Jeżeli zainstalowałeś PHP na swoim komputerze, gorąco zalecamy włączenie tej opcji na czas nauki (patrz dodatek A).
- Jeżeli w trakcie uruchamiania skryptu PHP zobaczysz pustą stronę, sprawdź również, czy nie ma błędów w kodzie HTML.
- Funkcja `ini_set()` może być wykorzystana do zmiany tylko niektórych ustawień. Więcej informacji znajdziesz w podręczniku PHP.
- Pamiętaj, że dyrektywa `display_errors` steruje tylko tym, czy komunikaty błędów będą wysyłane do przeglądarki. Nie tworzy ona błędów ani nie zapobiega ich powstawaniu.
- Zalecane jest, aby wyłączyć parametr `display_errors` w produkcyjnej witrynie WWW, ponieważ błędy są zarówno niewłaściwe, jak również mogą spowodować naruszenie systemu bezpieczeństwa. W trakcie tworzenia witryny powinieneś jednak włączyć parametr `display_errors`.
- Możliwe jest, że twórcy PHP zmienią inne ustawienia, które zmodyfikują sposób działania skryptów opisanych w tej książce. Niezbędne stanie się wtedy wprowadzenie odpowiednich korekt.

Raportowanie błędów

Drugim z trzech zagadnień konfiguracji PHP, o których powinieneś wiedzieć, jest sposób raportowania błędów. W PHP mamy osiem typów błędów oraz trzy typy definiowane przez użytkownika (o których nie będziemy tutaj mówić). W tabeli 3.1 wymienione są cztery najważniejsze typy błędów, wraz z opisami i przykładami.

Można określić, w jaki sposób PHP będzie raportował błędy, korzystając z funkcji `error_reporting()`. Wymieniona funkcja jako parametru oczekuje liczby lub stałej (stała to ciąg nie ujęty w apostrofy, który ma zdefiniowane wcześniej znaczenie) w celu zmiany tego poziomu. Na przykład:

```
error_reporting(0); // Wyłączamy całkowicie.
error_reporting(E_ALL); // Raportujemy wszystko.
error_reporting(E_ALL & ~E_NOTICE);
➥// Nie pokazujemy typu notice.
```

Ostatni przykład pozwala na ustawienie wyświetlania wszystkich komunikatów błędów poza typem `notice` (`&~` oznacza *i nie*). Można zastosować to ustawienie do naszego skryptu *obsluga_formularz.php*, aby usunąć komunikaty o błędach (rysunek 3.12).

Listing 3.6. *Można ustawić poziom raportowania błędów przez PHP, aby dawał mniej lub więcej informacji zwrotnej*

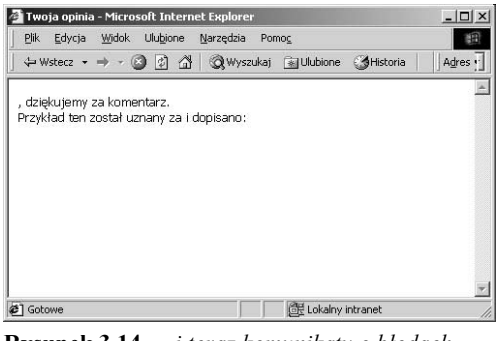
Listing

```
 1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
   1.0 Transitional//EN"
 2         "http://www.w3.org/TR/xhtml1/DTD/
           xhtml1-transitional.dtd">
 3 <html xmlns="http://www.w3.org/1999/xhtml">
 4 <head>
 5     <meta http-equiv="content-type"
       content="text/html;
       charset=iso-8859-2" />
 6     <title>Twoja opinia</title>
 7 </head>
 8 <body>
 9 <?php // Listing 3.6 obsluga_formularz.php
   (wersja trzecia)
10
11 ini_set ('display_errors', 1);
   // Pozwól mi uczyć się na błędach.
12 error_reporting(E_ALL & ~E_NOTICE);
   // Nie pokazujemy typu notice.
13
14 // Strona ta otrzymuje dane z opinia.html.
15 // Otrzymuje: tytuł, nazwę, e-mail,
   opinię oraz komentarz.
16
17 print "$tytul $imie, dziękujemy za
   komentarz. <br />";
18 print "Przykład ten został uznany za
   $opinia i dopisano: $komentarz";
19
20 ?>
21 </body>
22 </html>
```

Tabela 3.1. *Cztery najczęściej spotykane typy błędów w PHP*

Typy błędów PHP		
Typ	**Opis**	**Przykład**
Notice	Błąd niekrytyczny, który może (ale nie musi) oznaczać problem.	Odwołanie się do zmiennej bez wartości.
Warning	Błąd niekrytyczny, który nie zawsze oznacza wystąpienie poważnego problemu.	Przekazanie do funkcji parametru o niewłaściwym typie.
Parse error	Błąd krytyczny powodowany przez błąd składni.	Brak średnika lub brak pary dla apostrofu, nawiasu lub klamry.
Error	Ogólny błąd krytyczny.	Problem z przydziałem pamięci.

Rysunek 3.13. *Jeszcze raz wypełniamy formularz…*



Rysunek 3.14. *…i teraz komunikaty o błędach zniknęły*

Aby ustawić poziom raportowania błędów:

1. Otwórz w edytorze plik *obsluga_formularz.php* (listing 3.5).
2. Po wierszu zawierającym `ini_set()` wprowadź następujący wiersz (listing 3.6):

```
error_reporting(E_ALL & ~E_NOTICE);
```

3. Zapisz plik pod nazwą `obsluga_formularz.php`.
4. Wyślij plik na serwer WWW i obejrzyj go w przeglądarce (rysunki 3.13 oraz 3.14).

   W tym momencie powinieneś widzieć wynik pomyślnego przetworzenia skryptu (rysunek 3.9) lub brakujące wartości zmiennych, ale bez komunikatów błędów (rysunek 3.14). W następnym podrozdziale pokażemy, w jaki sposób poradzić sobie z problemem brakujących zmiennych.

Wskazówki

- W podręczniku PHP wymienione są wszystkie poziomy raportowania błędów, ale przytoczone tutaj są najbardziej użyteczne.
- Możesz również zmienić poziom raportowania błędów PHP w pliku *php.ini*, ale taka zmiana wpłynie na wszystkie skrypty. Jeżeli masz własny serwer PHP, w czasie tworzenia własnych skryptów prawdopodobnie będziesz chciał odpowiednio skonfigurować tę funkcję. Zapoznaj się z podrozdziałem „Konfigurowanie PHP” w dodatku A.
- W naszym skrypcie pod nazwą *obsluga_formularz.php* błędy `notice` wskazują na problem — do zmiennej nie została przypisana wartość. Ale ten problem jest na tyle oczywisty w wynikowej stronie WWW, że wyłączenie komunikatu o błędzie wydaje się całkowicie uzasadnione.