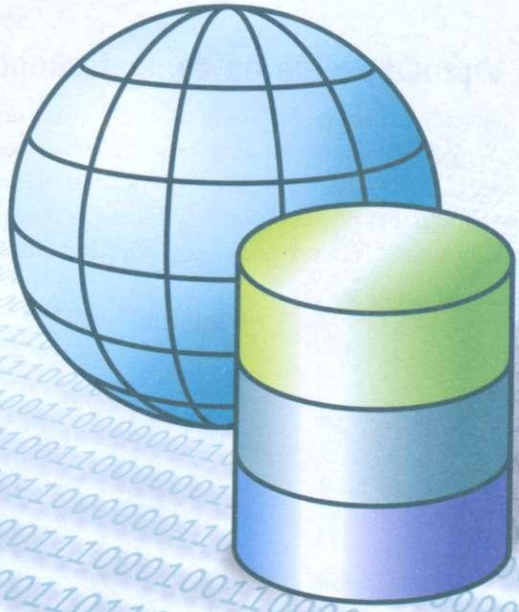
Aplikacje bazodanowe dostępne za pośrednictwem Sieci

VI

Rozdział



Temat 31. Budowanie interaktywnej witryny internetowej

Warto powtórzyć

Uwaga: W celu powtórzenia materiału skorzystaj z tematów A1, B10 i B11, *Informatyka podstawowa.*

1. Jakie poznałeś narzędzia do tworzenia stron internetowych?
2. Jak zbudowany jest adres URL?
3. W jaki sposób zapisana jest strona WWW?
4. Czym charakteryzuje się HTML? Czym są znaczniki?
5. W jaki sposób określamy kodowanie znaków w plikach HTML?
6. Czym różnią się skrypty działające po stronie serwera od skryptów działających po stronie przeglądarki?
7. Do czego służą arkusze stylów CSS?

1. Wprowadzenie do dynamicznego przetwarzania stron

Na lekcjach informatyki realizowanej w zakresie podstawowym tworzyliśmy strony WWW, korzystając ze znaczników HTML. Tworzone strony miały charakter statyczny – znaczniki HTML zapisane w pliku źródłowym trafiały do przeglądarki internetowej w niezmienionej postaci (rys. 1.). W celu zmiany treści strony wysyłanej do przeglądarki użytkownika należało zmodyfikować plik źródłowy strony.

Rys. 1. Schemat działania statycznej strony WWW

Strony dynamiczne są przed wysłaniem do użytkownika przetwarzane przez serwer WWW. Kod źródłowy takich stron, oprócz zwykłych znaczników HTML, zawiera także znaczniki z poleceniami intepretowanymi przez odpowiednie oprogramowanie na serwerze. Plik umieszczony na serwerze WWW, zawierający kombinację poleceń dla interpretera i znaczników HTML, nazywamy **skryptem**. Efektem działania skryptu jest dokument, który trafia do przeglądarki internetowej (rys. 2.). Warto zauważyć, że użytkownik nie może podejrzeć kodu źródłowego takiego skryptu w przeglądarce za pomocą opcji **Pokaż źródło** – do przeglądarki trafiają tylko wyniki działania skryptu.

Języki skryptowe stały się niezbędnym narzędziem, bez którego nie istniałby żaden duży serwis WWW. Pozwalają one na pobieranie i przetwarzanie danych z różnych miejsc, a następnie wysyłanie ich w postaci dokumentu HTML do użytkownika. Umożliwiają także interakcję z osobami odwiedzającymi serwis.

Serwer WWW

to program działający na serwerze, udostępniający użytkownikom pliki i dynamicznie generowane dane za pośrednictwem protokołu HTTP.

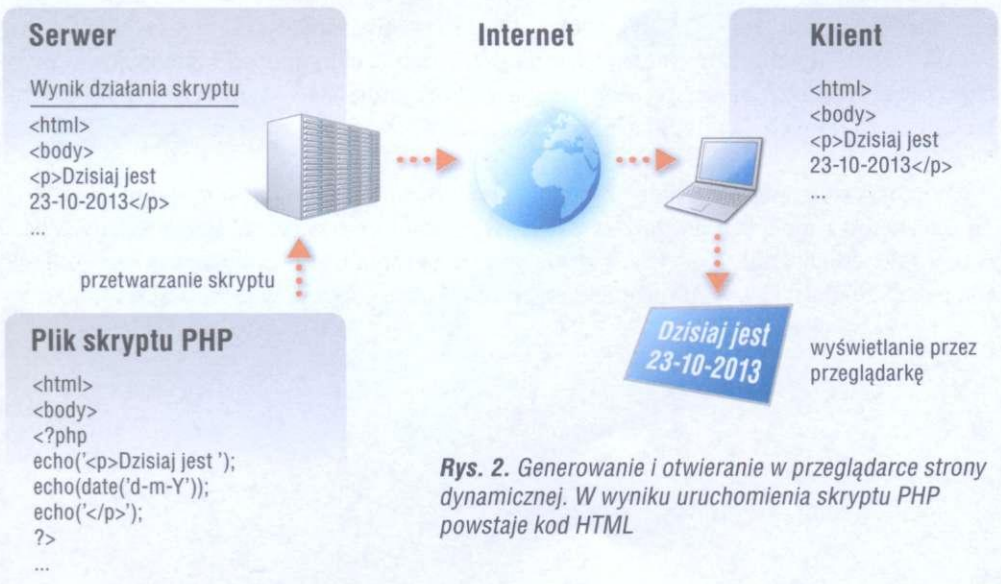
Skrypty wykonywane po stronie serwera najczęściej generują kod HTML. Mogą jednak generować dowolny typ danych – na przykład dokumenty PDF, pliki graficzne czy pliki tekstowe.

Język PHP jest stosunkowo łatwy do przyswojenia, a przy tym oferuje bardzo duże możliwości. Warto jednak wiedzieć, że istnieje wiele konkurencyjnych technologii, jak choćby ASP.Net, Java Server Pages (JSP), Ruby.

Przykład 1. Zastosowanie skryptu do wyszukiwania książek spełniających odpowiednie kryteria

Posiadamy bazę danych zawierającą informacje na temat zbioru książek. Możemy napisać skrypt działający na serwerze, który pozwoli wyszukiwać tytuły spełniające określone kryteria (na przykład utwory wybranego autora). Zadaniem skryptu będzie wysłanie zapytania do bazy danych, przetworzenie wyniku i wygenerowanie kodu HTML reprezentującego tabelę z listą książek.

Jednym z najczęściej stosowanych języków służących do pisania skryptów wykonywanych po stronie serwera jest język **PHP** (ang. *PHP Hypertext Preprocessor*).



Rys. 2. *Generowanie i otwieranie w przeglądarce strony dynamicznej. W wyniku uruchomienia skryptu PHP powstaje kod HTML*

Cechą stron generowanych dynamicznie po stronie serwera jest to, że dane trafiające do klienta mogą się zmieniać, mimo że plik źródłowy przechowywany na serwerze pozostaje niezmieniony. Na przykład w wyniku przetwarzania skryptu PHP pokazanego na rysunku 2. każdego dnia zostanie wyświetlona inna data.

Skrypty mogą zapisywać i pobierać dane z wielu różnych źródeł. Najczęściej do przechowywania danych stosuje się bazy danych lub tradycyjne pliki. Możliwa jest także komunikacja z innymi serwerami za pośrednictwem sieci. Dane mogą być także pobierane od użytkownika, na przykład za pomocą formularzy HTML.

Skrypty wykonywane na serwerze i skrypty uruchamiane przez przeglądarkę użytkownika wzajemnie się uzupełniają. Większość nowoczesnych serwisów stosuje oba rodzaje skryptów.

Skrypty PHP są wykonywane przez interpreter. Sprawdza on poprawność składniową kodu i wykonuje kolejne instrukcje.
Zaletą takiego rozwiązania jest łatwość dokonywania poprawek. Po zmianie kodu źródłowego nie trzeba kompilować skryptu, tak jak kompiluje się programy w językach Pascal, C++ czy Java. Przy kolejnym uruchomieniu skryptu PHP interpreter odczyta i zinterpretuje najnowszą wersję kodu źródłowego. Jest to ważna cecha, odróżniająca programy interpretowane od kompilowanych.

2. Konfigurowanie serwera Apache, interpretera PHP i bazy danych MySQL

Według danych Netcraft ze stycznia 2013 r. 55% serwisów WWW na świecie wykorzystywało serwer Apache.

Do uruchomienia strony zbudowanej w oparciu o skrypty PHP potrzebny jest odpowiednio przygotowany serwer WWW. Obecnie najpopularniejszy jest serwer Apache HTTP Server (często nazywany po prostu Apache), dostępny bezpłatnie na licencji o otwartym kodzie źródłowym. Serwer dostępny jest w wersjach dla wielu systemów operacyjnych, w tym Microsoft Windows, Linux i Apple OS X.

Większość dynamicznych stron wymaga także mechanizmu umożliwiającego przechowywanie danych na serwerze. Obecnie wykorzystuje się w tym celu najczęściej relacyjne bazy danych. Najpopularniejszą bazą danych wykorzystywaną razem z PHP jest MySQL.

Wykorzystywanie bazy danych do przechowywania informacji powiązanych ze stroną internetową może się początkowo wydawać skomplikowane. W większości zastosowań jest to jednak znacznie lepsze rozwiązanie niż gromadzenie danych w postaci plików. Baza danych ułatwia wyszukiwanie danych za pomocą zapytań w języku SQL.

Przykład 2. Zastosowanie skryptów do tworzenia bazy książek

Większość sklepów internetowych przechowuje w bazach danych informacje o dostępnych produktach, klientach i składanych przez nich zamówieniach. Wykorzystanie bazy danych pozwala na łatwe generowanie różnych zestawień – na przykład historii zamówień danego klienta bądź też raportu z listą klientów, którzy w ciągu ostatniego miesiąca złożyli zamówienia na określoną kwotę. W przypadku plików konieczne jest samodzielne napisanie kodu, który odnajdzie żądane informacje i połączy ze sobą dane zapisane w różnych miejscach.

Serwer bazy danych oferuje także mechanizmy, które ułatwiają przetwarzanie transakcji w bazie danych. Nowoczesne bazy danych pozwalają na równoległe modyfikowanie danych przez wielu użytkowników, nawet jeśli dane te znajdują się w tej samej tabeli. W przypadku zwykłych plików zapewnienie obsługi równoległych modyfikacji jest trudne – najczęściej, chcąc zmodyfikować plik, musimy zablokować możliwość zapisu do tego pliku przez inne programy. Większość baz danych zawiera także mechanizmy zapewniające integralność relacji pomiędzy tabelami.

2.1. Konfiguracja pakietu XAMPP

Samodzielna instalacja serwera Apache, interpretera PHP i serwera MySQL w systemie Windows jest czasochłonna i wymaga ręcznej edycji plików konfiguracyjnych, aby komponenty te poprawnie ze sobą współpracowały. W Internecie dostępnych jest wiele pakietów, w których wszystkie te programy zostały skonfigurowane do współpracy ze sobą i mogą zostać uruchomione bez konieczności instalacji każdego programu z osobna. Popularne pakiety tego typu to XAMPP, WampServer czy EasyPHP. Zawierają one serwer Apache, interpreter skryptów PHP wraz z najpopularniejszymi bibliotekami, serwer bazy danych MySQL oraz różne pomocnicze narzędzia.

W przykładach i ćwiczeniach tego rozdziału zastosowano pakiet XAMPP Portable Lite 1.8.1 w wersji dla systemu Windows. Najnowszą wersję pakietu można pobrać ze strony *http://www.apachefriends.org/en/xampp.html*. Pakietu XAMPP w wersji „Portable" nie trzeba instalować. Wystarczy wypakować archiwum na dysk twardy komputera lub dysk zewnętrzny (np. pendrive).

Ćwiczenie 1.

Wypakuj pakiet XAMPP do katalogu *C:\xampp* na dysku twardym komputera (lub, jeśli twoje konto nie posiada uprawnień do zapisu w tym folderze, do katalogu *xampp* w katalogu głównym innego dysku).

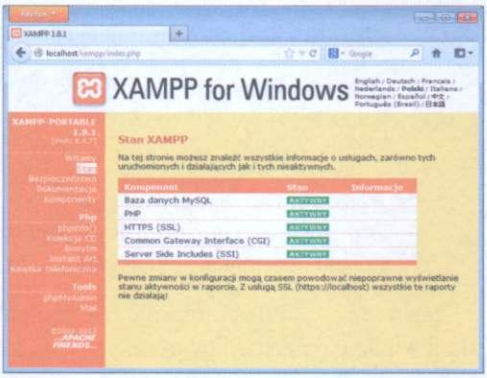
Wskazówka: Aby pakiet XAMPP mógł poprawnie działać bez konieczności dodatkowej konfiguracji, zalecane jest wypakowanie plików do podkatalogu „xampp" w głównym katalogu danego dysku (np. „C:\xampp\", „D:\xampp\", itd.).

W katalogu głównym XAMPP dla Windows znajduje się kilka skryptów, umożliwiających uruchamianie i zatrzymywanie poszczególnych komponentów. Najważniejsze z nich to:

apache_start.bat – uruchamia serwer Apache,
apache_stop.bat – zatrzymuje serwer Apache,
mysql_start.bat – uruchamia serwer baz danych MySQL,
mysql_stop.bat – zatrzymuje serwer MySQL.

Ćwiczenie 2.

Uruchom kolejno pliki *apache_start.bat* i *mysql_start.bat* z katalogu głównego XAMPP. Sprawdź w oknie terminala, czy programy poprawnie się uruchomiły. Jeżeli tak, uruchom przeglądarkę internetową i przejdź pod adres *http://localhost*. Powinna otworzyć się strona XAMPP. Po kliknięciu linku **Status** powinieneś zobaczyć stronę podobną do przedstawionej na rysunku 3. – informuje ona, że wszystkie komponenty zostały poprawnie uruchomione.



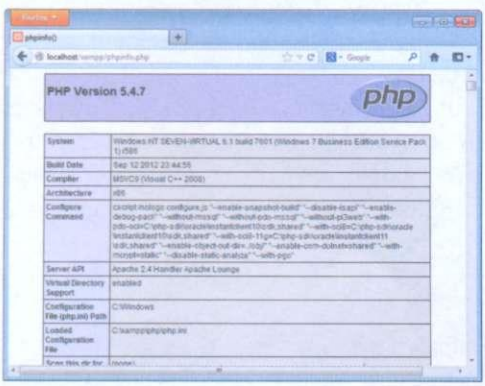
Rys. 3. *Strona prezentująca status poszczególnych komponentów pakietu XAMPP*

2.2. Pierwszy skrypt PHP

Strony WWW przechowywane są w podkatalogu *htdocs* w katalogu *xampp* (na przykład *C:\xampp\htdocs*).

Przykład 3. Wyświetlanie podsumowania konfiguracji PHP

W celu sprawdzenia poprawności instalacji serwera Apache i PHP można użyć następującego kodu PHP:



Rys. 4. *Strona testowa wyświetlająca informacje o aktualnie używanej wersji PHP*

```
<?php
phpinfo();
?>
```

Jeżeli umieścimy podany kod w pliku *phpinfo.php* i zapiszemy go w katalogu dokumentów serwera WWW (na przykład *C:\xampp\htdocs\phpinfo.php*), to po wpisaniu w przeglądarce internetowej na tym samym komputerze adresu *http://localhost/phpinfo.php* powinna wyświetlić się strona podobna do pokazanej na rysunku 4.

Uwaga: Warto wiedzieć, z której wersji PHP korzystamy. Niektóre skrypty działają bowiem tylko z określonymi wersjami interpretera. Powyższy skrypt testowy pozwala w łatwy sposób sprawdzić wersję PHP. Wyświetla także listę wszystkich dodatkowych modułów PHP zainstalowanych na serwerze. Do uruchamiania przykładów kodu z niniejszego podręcznika zalecamy użycie PHP w wersji 5.3 lub nowszej.

Należy pamiętać o nadawaniu tworzonym plikom odpowiedniego rozszerzenia. Rozszerzenie informuje serwer, jakiego typu dane zawiera plik. Dla skryptów PHP zalecane jest stosowanie rozszerzenia *php*.

Aby skrypt mógł zostać uruchomiony na serwerze poprzez odwołanie z przeglądarki użytkownika, należy nadać plikowi źródłowemu rozszerzenie *php* i skopiować plik do odpowiedniego katalogu, udostępnionego przez serwer WWW jako publiczny (podobnie jak publikuje się zwykłe pliki HTML). W przypadku XAMPP jest to podkatalog *htdocs* w katalogu głównym pakietu (na przykład *C:\xampp\htdocs*). Na serwerze musi być zainstalowany interpreter PHP.

Aby wykonać skrypt, należy otworzyć go w przeglądarce WWW, podając adres, pod jakim jest on dostępny. Na przykład, jeżeli umieścimy plik *skrypt.php* w katalogu *C:\xampp\htdocs\testy\skrypt.php* (lub analogicznym, w zależności od tego, gdzie znajduje się pakiet XAMPP), skrypt będzie dostępny pod adresem *http://localhost/testy/skrypt.php*.

2.3. Pobieranie dokumentacji PHP i MySQL

Podczas korzystania z PHP często będziemy sięgać do dokumentacji. Można ją przeglądać online lub pobrać na dysk twardy komputera ze strony internetowej *http://php.net/download-docs.php*. Dla systemu Windows dostępna jest częściowo spolonizowana dokumentacja w formacie CHM (jest to standardowy format plików **Pomocy** w systemach Windows) lub w postaci plików HTML. Dokumentacja omawia szczegółowo składnię języka, a także opisuje funkcje oferowane przez PHP i dołączone biblioteki.

Dokumentację serwera baz danych MySQL można pobrać na stronie *http://dev.mysql.com/doc/* (oficjalna dokumentacja dostępna jest tylko w języku angielskim). Dokumentacja opisuje szczegółowo obsługiwane przez serwer typy danych, składnię poleceń SQL oraz zagadnienia związane z konfiguracją serwera. Sposób łączenia się z bazą MySQL z poziomu PHP został opisany w kolejnym temacie.

3. Pisanie skryptów w języku PHP

3.1. Wyświetlanie danych instrukcją `echo`

Kod w języku PHP umieszcza się bezpośrednio w kodzie dokumentu HTML. Oddziela się go specjalnymi znacznikami: `<?php` oraz `?>`. Każda instrukcja, podobnie jak w językach Pascal, C++ czy Java, kończy się znakiem średnika.

Przykład 4. Kod prostego skryptu PHP, wyświetlającego bieżącą datę

```
<html>
<head>
<title>Skrypt wyświetlający bieżącą datę</title>
</head>
<body>
<h1>Witaj świecie!</h1>
<?php
echo('<p>Dzisiaj jest <b>'.date(,d-m-Y').'</b></p>');
?>
</body>
</html>
```

Rezultatem wykonania skryptu dnia 23 października 2013 roku jest kod HTML:

```
<html>
<head>
<title>Skrypt wyświetlający bieżącą datę</title>
</head>
<body>
<h1>Witaj świecie!</h1>
<p>Dzisiaj jest <b>23-10-2013</b></p></body>
</html>
```

Kod HTML (fragmenty pliku umieszczone poza znacznikami `<?php` i `?>`) przekazywany jest do przeglądarki w niezmienionej postaci. W miejscach, w których znajdują się znaczniki PHP, umieszczane są dane utworzone przez skrypt PHP – najczęściej będące rezultatem wykonania instrukcji `echo`.

Instrukcja `echo` dołącza do wynikowego kodu HTML łańcuch (ciąg znaków) podany jako parametr.

W skrypcie podanym w przykładzie 4. instrukcja `echo` generuje ciąg znaków „ Dzisiaj jest 23-10-2013 ", będący połączeniem trzech łańcuchów znakowych (kropka jest w PHP operatorem łączącym łańcuchy). Pierwszy i ostatni z tych łańcuchów to ciągi liter pomiędzy znakami apostrofu. Środkowy ciąg jest natomiast rezultatem działania funkcji `date`, dla której jako parametr podano format, w jakim ma zostać wypisana data.

Fragmenty strony generowane za pomocą PHP są traktowane przez przeglądarkę tak samo jak statyczne fragmenty kodu HTML. Przeglądarka zinterpretuje więc wygenerowany znacznik akapitu `<p>`, a także pogrubi tekst zawarty pomiędzy znacznikami `<b>...</b>`.

3.2. Kodowanie UTF-8

Aby kod tworzonych dynamicznie stron mógł poprawnie współpracować z bazą danych i obsługiwać znaki w wielu różnych alfabetach (w tym znaki charakterystyczne dla języka polskiego), we wszystkich kolejnych skryptach będziemy stosować kodowanie UTF-8. W tym celu w sekcji `<head>` tworzonych dokumentów będziemy umieszczać znacznik: `<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">`.

Jeżeli do tworzenia skryptów będziemy używać programu Notatnik, konieczne będzie wybranie kodowania UTF-8 przy zapisywaniu pliku w oknie **Zapisz jako** (domyślnie wybierane jest kodowanie ANSI, oznaczające w polskich systemach Windows stronę kodową Windows-1250).

Ćwiczenie 3.

Zmodyfikuj kod źródłowy przykładowego skryptu z przykładu 4, dodając do niego nagłówek określający kodowanie UTF-8. Zapisz plik pod nazwą *T31_cw3.php*, używając kodowania UTF-8, i umieść go w odpowiednim katalogu dla serwera WWW. Otwórz skrypt w przeglądarce internetowej, aby sprawdzić, czy polskie znaki zostaną poprawnie wyświetlone.

Wskazówka: Plik o nazwie *T31_cw3.php* można otworzyć, wpisując w pasku adresu *http://localhost/T31_cw3.php*.

3.3. Stosowanie zmiennych i operatorów

Język PHP, w przeciwieństwie do języków takich, jak Pascal, C++ czy Java, nie nakłada na programistę wymogu deklarowania zmiennych przed ich użyciem. W PHP, aby utworzyć zmienną, po prostu przypisuje się jej wartość.

Zmienne oznacza się, umieszczając znak „$" przed ich nazwą. Nie ma potrzeby deklarowania typu zmiennej. Interpreter rozpoznaje go automatycznie na podstawie wartości przypisywanej zmiennej. Operatorem przypisania jest znak „=" (a nie „:=", jak w języku Pascal). W nazwach zmiennych rozróżniane są wielkie i małe litery.

Przykład 5. Korzystanie ze zmiennych i operatorów

Skrypt prezentuje wykorzystanie zmiennych w PHP oraz operatorów matematycznych i operatora łączenia łańcuchów znakowych.

```
<?php
?>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Wykorzystanie zmiennych i operatorów w PHP</title>
</head>
<body>
<?php
// zmienna $dzialanie będzie zawierać łańcuch znaków z tekstowym
// zapisem działania
$dzialanie = '(34 + 1) / 5 = ';
// zmienna $wynik będzie zawierać obliczony wynik działania
// w postaci liczby całkowitej
$wynik = (34 + 1) / 5;
echo('<p>Proste działanie matematyczne: '.$dzialanie.$wynik.'</p>');
?>
</body>
</html>
```

W przykładzie 5. zmienna *$dzialanie* przechowuje łańcuch znaków z tekstowym zapisem działania. Łańcuchy znaków powinny być otoczone znakami apostrofu lub cudzysłowu (wyjaśnienie różnic pomiędzy tymi dwoma zapisami można znaleźć w dokumentacji PHP).

W PHP do łączenia łańcuchów (ciągów znakowych), inaczej niż w języku Pascal, nie można stosować operatora „+". Służy on tylko do dodawania liczb. Wyrażenie '14' + '10' zwraca wartość 24, a nie '1410' (łańcuchy znaków '14' i '10' konwertowane są na liczby, których wartości są do siebie dodawane). Aby otrzymać efekt połączenia łańcuchów, należy użyć operatora „." (kropka).

Ćwiczenie 4.

Napisz krótki skrypt, który zaprezentuje różnice w działaniu pomiędzy operatorami „." (łączenie łańcuchów znakowych) i „+" (dodawanie liczb).

PHP, tak jak większość języków programowania, pozwala na stosowanie komentarzy. Ułatwiają one programiście zinterpretowanie przeznaczenia poszczególnych fragmentów kodu. Gdy w przyszłości zajdzie potrzeba modyfikacji skryptu, dużo łatwiej będzie przypomnieć sobie, za co odpowiadały poszczególne fragmenty.

Najczęściej stosuje się komentarze jednoliniowe, rozpoczynające się od znaków „//" i kończące wraz z przejściem do nowej linii. W przypadku konieczności umieszczenia dłuższych opisów używa się symboli „/*" i „*/" dla oznaczenia odpowiednio początku i końca komentarza. Treść komentarzy powinna być umieszczona wewnątrz znaczników PHP (pomiędzy `<?php` i `?>`).

Ćwiczenie 5.

Dołącz do pliku utworzonego w ćwiczeniu 4. krótki komentarz opisujący jego działanie.

4. Przesyłanie danych za pomocą formularzy HTML

Jedną z podstawowych funkcji języków skryptowych jest odbieranie i przetwarzanie danych przekazywanych przez użytkowników. Do przekazywania danych do serwera WWW wykorzystuje się między innymi formularze HTML. Spełniają one funkcje podobne do formularzy w bazach danych.

Formularze można stosować, na przykład, do:

- przekazywania wyszukiwarce haseł do znalezienia,
- przekazywania danych niezbędnych do zalogowania się w serwisie,
- dodawania wpisu do księgi gości,
- wyboru języka, w jakim wyświetlany będzie serwis,
- przeprowadzania ankiet.

Dane wprowadzone przez użytkownika w formularzu są przekazywane do wskazanego skryptu na serwerze (najczęściej to ten sam serwer, na którym znajduje się strona WWW zawierająca formularz).

Skrypt przetwarza odebrane dane i wykonuje niezbędne operacje – na przykład aktualizuje informacje w bazie danych, wysyła e-mail z powiadomieniem do administratora strony. Następnie skrypt generuje stronę wynikową bądź przekierowuje użytkownika do innej strony.

Aby utworzyć formularz, wystarczy do kodu HTML dodać znacznik:
`<form action="..." method="post">...</form>`,
podając jako wartość atrybutu `action` adres skryptu, który zajmie się obsługą danych wprowadzonych przez użytkownika.

Pomiędzy znacznikami `<form>` i `</form>` można umieszczać znaczniki opisujące elementy formularza, takie jak pola tekstowe, przyciski, pola wyboru, listy. Przykłady pól przedstawia tabela 1. Każdemu elementowi formularza przypisujemy unikalną nazwę, która pozwoli odczytać jego wartość w skrypcie odbierającym dane. Na przykład znacznik `<input type="text" name="nazwisko">` definiuje pole tekstowe o nazwie *nazwisko*. Warto zauważyć, że nazwy pól nie są wyświetlane przez przeglądarkę. Aby formularz był czytelny, należy samodzielnie zadbać o umieszczenie odpowiednich opisów obok pól.

Znacznik	Element
`<input type="text" name="pole1">`	Pole tekstowe pole tekstowe
`<textarea name="pole2"></textarea>`	Wieloliniowe pole tekstowe wieloliniowe pole tekstowe

`<select name="lista1">`
`<option value="k">Kobieta</option>`
`<option value="m">Mężczyzna</option>`
`</select>`
	Lista wyboru
	Kobieta

`<input name="wybor1" type="checkbox" value="tak">Tak, chcę się zapisać` `</input>`	Pole wyboru ☐ Tak, chcę się zapisać
`<input type="submit" value="Wyślij dane">`	Przycisk, którego naciśnięcie spowoduje wysłanie danych wprowadzonych w formularzu do serwera Wyślij dane

***Tabela 1.** Przykładowe pola, które można umieścić w formularzu*

Przykład 6. Kod strony wyświetlającej prosty formularz i odbierającej dane z formularza

Przyjęliśmy, że skrypt obsługujący formularz będzie umieszczony w pliku o nazwie *formularz.php* (parametr `action` znacznika `form`).

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Obsługa prostego formularza</title>
</head>
<body>
<?php
if(empty($_POST['imie'])) {
  // nie przesłano danych lub pole "imie" jest puste - wyświetlamy
  // formularz
  ?>
  <form action="formularz.php" method="post">
    Podaj imię: <input type="text" name="imie" value=""><br>
    <input type="submit" value="Wyślij">
  </form>
  <?php
} else {
  // otrzymano wartość "imie" z formularza - wyświetlamy powitanie
  echo('<h1>Witaj '.htmlspecialchars($_POST['imie']).'!</h1>');
  echo('<p>Poprawnie odebrano dane z formularza.</p>');
} // koniec warunkowego wyświetlania formularza
?>
</body>
</html>
```

Rys. 5. Prosty formularz otwarty w przeglądarce internetowej

Po kliknięciu przycisku *Wyślij* dane z formularza zostaną wysłane do serwera. Do obsługi formularza wykorzystujemy ten sam skrypt. Instrukcja warunkowa **`if`** rozpoznaje, czy dane zostały przesłane. Dane przesyłane metodą POST są automatycznie umieszczane w tablicy *$_POST*, której indeksami są nazwy pól formularza (w języku PHP indeksami elementów w tabeli mogą być nie tylko liczby, ale także ciągi znaków). Wartość pola *imie* odczytujemy za pomocą odwołania `$_POST['imie']`. Funkcja `empty()` sprawdza, czy przekazana wartość nie jest pusta.

Rys. 6. Schemat działania formularzy

Ćwiczenie 6.

Rozbuduj formularz z przykładu 6. o dodatkowe pola, wykorzystując elementy formularza zaprezentowane w tabeli 1. Zmodyfikuj kod odbierający dane z formularza, tak aby uwzględniał dodane pola.

Wskazówka: Pamiętaj o zapisaniu pliku pod nazwą *formularz.php*.

Warto zapamiętać

- Omówiliśmy dwa rodzaje stron: statyczne (zawierające tylko kod HTML) i dynamiczne (zawierające oprócz kodu HTML polecenia skryptowe wykonywane przez odpowiednie oprogramowanie na serwerze).
- Skrypty przetwarzane są przez interpreter. Dlatego skryptów PHP nie trzeba kompilować. Aby zmodyfikować działanie skryptu, wystarczy zmodyfikować jego kod źródłowy.
- Polecenia PHP umieszcza się pomiędzy znacznikami `<?php` i `?>`.
- Aby wykonać skrypt, należy go najpierw umieścić w odpowiednim katalogu na serwerze, a następnie otworzyć w przeglądarce.

- Do wypisywania danych służy w PHP instrukcja `echo`.
- W PHP nie ma potrzeby deklarowania zmiennych. W celu utworzenia zmiennej przypisuje się jej wartość.
- Formularze na stronach internetowych umożliwiają przekazywanie danych do serwera WWW. Wykorzystywane są np. podczas logowania się w serwisie, wypełniania ankiety, dodawania wpisu do księgi gości, głosowania w sondażu, jak również podczas szukania informacji, na przykład gdy wprowadzamy hasła do wyszukiwarki.

Pytania, problemy

1. Jak nazywa się najpopularniejsze obecnie oprogramowanie serwera WWW?
2. Czym jest interpreter skryptów?
3. Jakie rozszerzenie powinien mieć plik zawierający skrypt PHP?
4. Jaka jest przewaga wykorzystania bazy danych do przechowywania danych serwisu w porównaniu z bezpośrednim zapisywaniem danych w plikach?
5. W jaki sposób należy wykonać napisany skrypt, aby sprawdzić jego działanie?
6. Dlaczego istotne jest opisywanie tworzonego kodu za pomocą komentarzy? W jaki sposób oznaczamy komentarze w języku PHP?
7. W jaki sposób tworzy się zmienne w PHP? Podaj przykłady przypisywania do zmiennych wartości liczbowych i ciągów znaków.
8. Jaki wynik otrzymamy, uruchamiając poniższy skrypt?

```
<?php
$name = 'Kacper';
$Name = 'Krajewski';
echo($name.'  '.$Name);
?>
```

9. Wyjaśnij różnicę pomiędzy operatorami „.” oraz „+”.
10. Jakich znaczników używa się do tworzenia formularzy w języku HTML? Jakie elementy może zawierać formularz?
11. W jaki sposób odwołujemy się w PHP do danych przesłanych z formularza?

Zadania

1. Napisz skrypt, który wyliczy i wyświetli listę kolejnych potęg dwójki (od 2^1 do 2^{50}).
   Wskazówka: Wykorzystaj pętlę `for`:

```
<?php
for($wykladnik=1; $wykladnik<=50; $wykladnik++) {
  echo 'Aktualna wartość wykładnika: '.$wykladnik.'<br>';
}
?>
```

2. Napisz skrypt obliczający stan konta bankowego w kolejnych pięciu latach, przyjmując, że na koncie było początkowo 1000 zł, a oprocentowanie wynosi 10% w stosunku rocznym (odsetki kapitalizowane są co roku).
3. Wyszukaj w dokumentacji PHP opis funkcji `date`. Napisz skrypt wypisujący aktualną datę i godzinę.
4. Napisz skrypt, który będzie wypisywał iloczyn dwóch liczb całkowitych, wprowadzonych w polach formularza HTML.
5. Zaprojektuj formularz, w którym użytkownik będzie podawał informacje na swój temat (np. imię, nazwisko, wiek, płeć, hobby). Wykorzystaj elementy takie, jak listy rozwijane czy przyciski wyboru. Napisz skrypt, który będzie odbierał i wyświetlał dane z formularza.

Dla zainteresowanych

6. Korzystając z opisu funkcji `date` w dokumentacji PHP, napisz skrypt wyświetlający bieżącą datę z miesiącem i dniem tygodnia w formie słownej (piątek, 17 października 2003).
   Wskazówka: Wykorzystaj następujący fragment kodu (w razie potrzeby zmodyfikuj go):

```
<?php
// deklaracja tablicy z nazwami dni tygodnia (indeksami elementów
są liczby od 0 do 6)
$dniTygodnia = array("niedziela", "poniedziałek", "wtorek",
"środa", "czwartek", "piątek", "sobota");
// pobranie numeru dnia tygodnia
$numerDniaTygodnia = date("w");
// wyświetlenie nazwy dnia tygodnia pobranej z tablicy
echo($dniTygodnia[$numerDniaTygodnia]);
?>
```

   Dlaczego pierwszym elementem tablicy jest „niedziela"?
7. Sprawdź adres IP komputera, na którym uruchomiony jest XAMPP (na przykład, wpisując w linii komend systemu Windows polecenie *ipconfig*). Spróbuj połączyć się z serwerem HTTP, działającym na innym komputerze w tej samej sieci lokalnej. W przypadku błędów połączenia sprawdź ustawienia zapory sieciowej (firewalla) w systemie Windows.
8. Rozbuduj skrypt podany w przykładzie 6, tak aby rozpoznawał płeć użytkownika na podstawie wpisanego imienia i wyświetlał dodatkową informację (na przykład: „Płeć rozpoznana na podstawie imienia: mężczyzna").
   Wskazówka: Przyjmij, że wszystkie polskie imiona żeńskie kończą się na literę „a". Możesz też uwzględnić imiona będące wyjątkiem od tej zasady, np. męskie imię Barnaba. Ostatnią literę ciągu znaków możesz odczytać, używając wyrażenia `mb_substr($imie, -1)`.

Temat 32. Witryny internetowe oparte na bazach danych

1. **Korzystanie z baz danych z poziomu PHP**
   - 1.1 Tworzenie konta użytkownika i bazy danych na serwerze MySQL
   - 1.2 Łączenie się z bazą MySQL z poziomu PHP
   - 1.3 Wykonywanie zapytań do bazy danych z poziomu PHP
2. **Tworzenie prostej księgi gości**
   - 2.1 Tworzenie tabeli na wpisy z księgi gości
   - 2.2 Dodawanie wpisów do księgi gości za pomocą instrukcji `INSERT`
   - 2.3 Tworzenie formularza dodającego wpisy do bazy danych
   - 2.4 Odczytywanie danych z bazy za pomocą instrukcji `SELECT`

Warto powtórzyć

1. Jakie są cechy relacyjnej bazy danych? W jaki sposób zorganizowane są dane w takiej bazie?
2. Jaką rolę pełni klucz podstawowy w tabeli?
3. Czym jest SQL?
4. Do czego służy instrukcja `SELECT` w języku SQL?
5. W jaki sposób uruchamia się serwer Apache i bazę danych MySQL w ramach pakietu XAMPP dla systemu Windows?
6. Jak przygotowuje się formularze w języku HTML? W jaki sposób odbiera się dane z formularzy w PHP?
7. Jakie są cechy programowania obiektowego? Przypomnij pojęcia: *obiekt*, *pole*, *metoda*.

1. Korzystanie z baz danych z poziomu PHP

PHP umożliwia łączenie się z wieloma popularnymi serwerami baz danych. W ramach niniejszego tematu omówimy łączenie się z bazą danych MySQL, powszechnie stosowaną do tworzenia dynamicznych stron internetowych.

Aby pobrać dane z bazy danych, należy:

- otworzyć połączenie do serwera baz danych (podając dane niezbędne do uwierzytelnienia połączenia),
- wskazać bazę danych, z którą będziemy pracować,
- wykonać zapytania SQL konieczne do pobrania danych,
- zamknąć połączenie z bazą (w przypadku języków skryptowych odbywa się to często automatycznie – po zakończeniu działania skryptu).

Do łączenia się z bazą danych MySQL użyjemy sterownika *pdo_mysql* dystrybuowanego wraz z PHP. Jest on domyślnie aktywny w ramach pakietu XAMPP.

1.1. Tworzenie konta użytkownika i bazy danych na serwerze MySQL

Zanim możliwe będzie połączenie się z bazą MySQL z poziomu PHP, konieczne jest utworzenie bazy danych i konta użytkownika na serwerze MySQL.

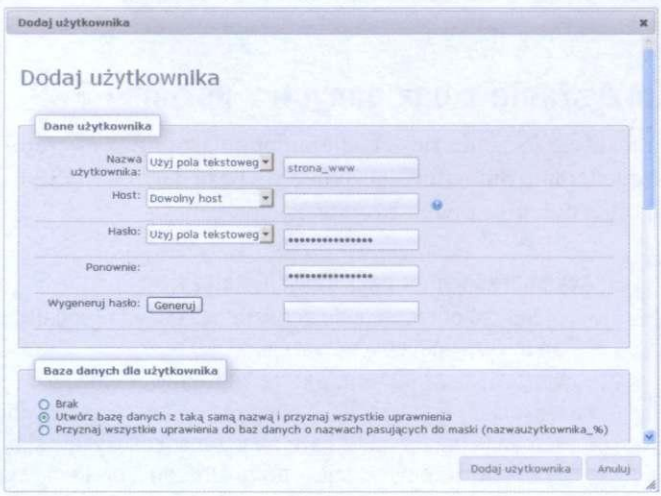
Pakiet XAMPP zawiera narzędzie o nazwie phpMyAdmin. Aby skorzystać z tego narzędzia, należy uruchomić serwery Apache i MySQL (sposób ich uruchamiania został opisany temacie 31.), a następnie otworzyć w przeglądarce stronę *http://localhost/phpmyadmin/*.

Aby móc zarządzać bazami danych i kontami użytkowników na serwerze, należy zalogować się na koncie o nazwie *root*. Jeżeli wcześniej nie ustawiliśmy hasła dla tego konta, pole z hasłem na stronie logowania wyświetlanej przez phpMyAdmin należy pozostawić puste.

Przykład 1. Tworzenie konta użytkownika MySQL i bazy danych

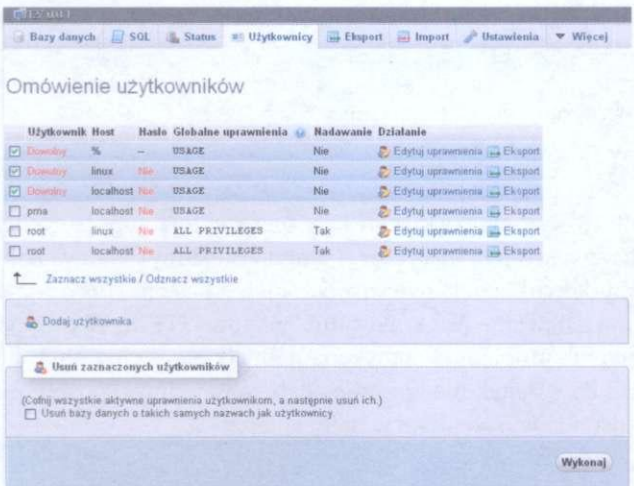
Aby za pomocą narzędzia phpMyAdmin utworzyć nowe konto użytkownika i bazę danych w MySQL, należy wykonać następujące kroki:

1. Zalogować się na stronie *http://localhost/phpmyadmin* na koncie *root*.
2. Przejść do zakładki **Użytkownicy** (*Users*) i kliknąć odnośnik **Dodaj użytkownika** (*Add user*).
3. W wyświetlonym formularzu wpisać nazwę i hasło dla nowego konta użytkownika. W dalszych przykładach będziemy posługiwać się kontem o nazwie *strona_www* z hasłem *HasloBazyDanych*.
4. W sekcji **Baza danych dla użytkownika** (*Database for user*) zaznaczyć opcję **Utwórz bazę danych z taką samą nazwą i przyznaj wszystkie uprawnienia** (*Create database with same name and grant all privileges*). Spowoduje to, że wraz z kontem użytkownika zostanie utworzona baza danych o takiej samej nazwie. Utworzone konto użytkownika będzie miało pełne uprawnienia do odczytu i zapisu danych w ramach utworzonej bazy.
5. Aby zatwierdzić tworzenie nowego konta, należy kliknąć przycisk **Dodaj użytkownika** (*Add user*) w dolnej części formularza.



***Rys. 1.** Formularz dodawania nowego konta użytkownika MySQL w narzędziu phpMyAdmin*

6. Konieczne jest także usunięcie części domyślnych wpisów dla kont użytkowników, które zostały automatycznie dodane przez XAMPP. W zakładce **Użytkownicy** (*Users*) należy zaznaczyć pola wyboru dla wszystkich wierszy tabeli, w których w polu **Użytkownik** (*User*) znajduje się napis **Dowolny** (*Any*) – rysunek 2. Następnie należy kliknąć przycisk **Wykonaj** (*Go*) w dolnej części ekranu, aby usunąć te wpisy. Pominięcie tego kroku może spowodować, że nie będzie możliwe zalogowanie się na dodanym przed chwilą koncie.



***Rys. 2.** Usuwanie domyślnych wpisów dla kont użytkowników dodanych przez phpMyAdmin*

Ćwiczenie 1.

Za pomocą narzędzia phpMyAdmin utwórz konto użytkownika i bazę danych o nazwie *strona_www* – zgodnie z instrukcją podaną w przykładzie 1. Następnie wyloguj się z phpMyAdmin i spróbuj zalogować się na dodanym przed chwilą koncie (przycisk umożliwiający wylogowanie się znajdziesz w lewym górnym rogu ekranu phpMyAdmin – ikona uchylonych drzwi).

1.2. Łączenie się z bazą MySQL z poziomu PHP

Łączenie się z bazą danych będzie wykorzystywane w wielu tworzonych przez nas skryptach. Dlatego wydzielimy osobny skrypt tworzący połączenie do bazy. Będziemy dołączać go do wszystkich skryptów odwołujących się do bazy danych za pomocą wbudowanej w PHP instrukcji `require`. Zaletą takiego rozwiązania jest uniknięcie duplikowania kodu – w przypadku konieczności dokonania poprawek lub zmiany konfiguracji połączenia wystarczy wprowadzić zmiany w jednym pliku.

Uwaga: W publicznie dostępnych serwisach **wyświetlanie wyjątków** przez moduł PDO łączący się z bazą danych powinno zostać wyłączone. W przeciwnym razie w przypadku wystąpienia błędu mogą zostać ujawnione wrażliwe dane, na przykład hasło dostępu do bazy danych.

Przykład 2. Skrypt tworzący połączenie do bazy danych

```
<?php
$DB = new PDO(
  // adres serwera bazy danych i nazwa bazy danych
  'mysql:host=localhost;dbname=strona_www',
  // nazwa konta użytkownika posiadającego dostęp do bazy
  'strona_www',
  // hasło dostępu
  'HasloBazyDanych',
  // kodowanie znaków dla połączenia i tryb obsługi błędów
  array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES 'UTF8'",
    PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION)
);
?>
```

Skrypt z przykładu 2. tworzy obiekt klasy PDO, reprezentujący połączenie do bazy danych. Adres serwera bazy danych to *localhost* (komputer lokalny). Nazwa bazy danych to *strona_www* (konto użytkownika o nazwie *strona_www* z hasłem *HasloBazyDanych*). Przy tworzeniu połączenia ustawiane jest kodowanie znaków UTF-8. W przypadku wystąpienia błędu połączenia lub problemów z wykonaniem jakiegoś zapytania, wygenerowany zostanie wyjątek. O ile wyjątek nie zostanie obsłużony w inny sposób, informacja o wystąpieniu wyjątku zostanie wyświetlona w przeglądarce.

Ćwiczenie 2.

Utwórz osobny plik *baza-danych.php*, zawierający kod używany do tworzenia połączenia z bazą danych. Skorzystaj ze wzoru pokazanego w przykładzie 2. W kolejnych przykładach będziemy się odwoływać do tego pliku za pomocą instrukcji require:

```
<?php
// wczytanie skryptu tworzącego połączenie do bazy danych
require('baza-danych.php');
?>
```

Uwaga: Jeżeli bezpośrednio w przeglądarce wpiszemy adres skryptu *baza-danych.php*, do przeglądarki zostanie wysłana pusta strona (o ile nie wystąpi błąd przy łączeniu się z bazą danych). Oznacza to, że połączenie zostało poprawnie utworzone.

Korzystając z pakietu XAMPP, będziemy używać bazy danych MySQL uruchomionej na tym samym komputerze co serwer WWW i interpreter PHP. W dużych serwisach WWW najczęściej jednak serwer WWW i serwer baz danych uruchomione są na osobnych komputerach. Komunikacja pomiędzy PHP i bazą danych odbywa się za pomocą sieci (najczęściej w oparciu o protokół TCP/IP). Baza danych może być współdzielona przez wiele serwerów korzystających z niej w tym samym czasie.

1.3. Wykonywanie zapytań do bazy danych z poziomu PHP

Proste zapytania, niewymagające dodawania parametrów, możemy wykonywać za pomocą metody `$DB->query()`, wywoływanej dla utworzonego obiektu bazy danych. W przykładzie 3. wykorzystamy polecenie języka SQL o nazwie `SHOW TABLES`. Pobiera ono listę tabel z bazy danych. Wyniki zapytania przetwarzamy w pętli `foreach`. Pętla umieszcza

kolejno wiersze z wyników zapytania w zmiennej `$row` i wykonuje dla nich kod zawarty w nawiasach klamrowych (w tym przypadku jest to instrukcja `echo`, wyświetlająca wartość pierwszego pola z każdego rekordu – pola w tablicy `$row` indeksowane są od 0).

Przykład 3. Pobranie listy tabel z bazy danych

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Lista tabel w bazie danych</title>
</head>
<body>
<h1>Lista tabel w bazie danych</h1>
<?php
// wczytanie skryptu tworzącego połączenie do bazy danych
require('baza-danych.php');
// pobranie i wyświetlenie listy tabel w bazie danych
foreach($DB->query('SHOW TABLES') as $row) {
  echo($row[0].'<br>');
}
?>
</body>
</html>
```

Skrypt z przykładu 3. wykorzystuje instrukcję `require` do wczytania utworzonego w przykładzie 2. pliku z konfiguracją połączenia do bazy danych MySQL.

Ćwiczenie 3.

Za pomocą interfejsu phpMyAdmin utwórz tabelę o nazwie *test* w bazie danych *strona_www*. Następnie uruchom skrypt z przykładu 3. i sprawdź, czy utworzona tabela pojawi się w wynikach działania skryptu.

Wskazówka: Aby dodać tabelę za pomocą phpMyAdmin, należy przejść do zakładki **Bazy danych** (*Databases*) i wybrać bazę danych, do której chcemy dodać tabelę. Następnie należy wypełnić formularz **Utwórz tabelę** (*Create table*) na dole strony, podając nazwę nowej tabeli i liczbę kolumn. Na kolejnym ekranie konieczne będzie jeszcze określenie definicji poszczególnych kolumn (do przykładowej tabeli wystarczy dodać pojedyncze pole *Id_rekordu* typu INT).

2. Tworzenie prostej księgi gości

Księga gości to element wielu stron internetowych. W dalszej części tematu pokażemy, jak zbudować prostą księgę gości w oparciu o PHP i bazę danych MySQL. Księga gości będzie się składać z dwóch podstron:

- skryptu wyświetlającego formularz dodawania nowych wpisów, obsługującego odbieranie danych z formularza i zapisywanie ich w bazie danych,
- skryptu wyświetlającego listę wpisów w księdze gości.

2.1. Tworzenie tabeli na wpisy z księgi gości

Do przechowywania wpisów z księgi gości utworzymy tabelę *ksiega_gosci_wpisy* w ramach dodanej wcześniej bazy danych *strona_www*.

Przykład 4.

Narzędzie phpMyAdmin umożliwia tworzenie tabel w bazie danych MySQL za pomocą graficznego interfejsu bądź też poprzez wprowadzanie poleceń języka SQL. Aby utworzyć nową tabelę za pomocą polecenia SQL, należy zalogować się w phpMyAdmin i wybrać z listy po lewej stronie żądaną bazę danych. Następnie należy kliknąć zakładkę **SQL**. Na potrzeby księgi gości można utworzyć tabelę, wykonując następujące polecenie:

```
CREATE TABLE ksiega_gosci_wpisy (
Id_wpisu INT UNSIGNED NOT NULL AUTO_INCREMENT,
Imie VARCHAR(100) NOT NULL,
Email VARCHAR(100) DEFAULT NULL,
Wpis TEXT NOT NULL,
Czas_dodania TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP,
PRIMARY KEY (Id_wpisu),
UNIQUE(Wpis(200)));
```

Polecenie: `CREATE TABLE nazwa (...)` tworzy nową tabelę o podanej nazwie. W nawiasach umieszcza się listę pól i kluczy.

Utworzona w przykładzie 4. tabela *ksiega_gosci_wpisy* będzie zawierać następujące pola:

- *Id_wpisu* – unikalny numer identyfikujący wpis (jeżeli przy dodawaniu rekordu nie określimy wartości dla tego pola, zostanie automatycznie wygenerowany kolejny numer rekordu),
- *Imie* – ciąg znaków o maksymalnej długości 100 znaków (nie dopuszczamy wartości `NULL`),
- *Email* – ciąg znaków o maksymalnej długości 100 znaków lub wartość `NULL`,
- *Wpis* – tekst (o maksymalnej długości około 64 kB),
- *Czas_dodania* – znacznik czasu (data i godzina).

Pole *Id_wpisu* zostało oznaczone jako klucz podstawowy (klauzula `PRIMARY KEY(Id_wpisu)`). Dodajemy także klucz `UNIQUE(Wpis(200)`. Wymusza on, aby wartość pola *Wpis* była unikalna dla każdego dodawanego rekordu (liczba w nawiasie oznacza, że przy sprawdzaniu unikalności porównywanych będzie pierwszych 200 znaków każdego wpisu). Klucz ten stanowi zabezpieczenie przed wielokrotnym dodaniem takiego samego wpisu.

Ćwiczenie 4.

Używając polecenia SQL z przykładu 4, dodaj do bazy danych o nazwie *strona_www* tabelę do przechowywania wpisów z księgi gości.

Uwaga: Pamiętaj, że wprowadzane polecenia SQL powinny kończyć się znakiem średnika. W przypadku pojedynczego polecenia SQL, wprowadzanego w phpMyAdmin, pojedynczy średnik zostanie automatycznie uzupełniony, ale przy większej liczbie poleceń brak średników spowodowałby wygenerowanie błędu.

2.2. Dodawanie wpisów do księgi gości za pomocą instrukcji `INSERT`

W języku SQL do dodawania nowych wpisów do tabeli służy instrukcja `INSERT`.

Przykład 5. Dodawanie wpisu do księgi gości

Aby dodać nowy wpis do tabeli *ksiega_gosci_wpisy*, można skorzystać z następującego polecenia SQL:

```
INSERT INTO ksiega_gosci_wpisy (Imie, Email, Wpis, Czas_dodania)
VALUES ('Tomek', 'tomek@example.com', 'Witam, to jest pierwszy wpis
w księdze gości', NOW());
```

W nawiasach po nazwie tabeli umieszczona jest lista pól, dla których będziemy podawać wartości. Celowo pominęliśmy pole *Id_wpisu* – wartość dla tego pola zostanie wygenerowana automatycznie. Dla pozostałych pól samodzielnie określamy wartości, podając ich listę w nawiasach okrągłych po słowie kluczowym `VALUES`. Dla pola *Czas_dodania* wstawiamy aktualną datę i czas za pomocą funkcji SQL o nazwie `NOW()`.

Ćwiczenie 5.

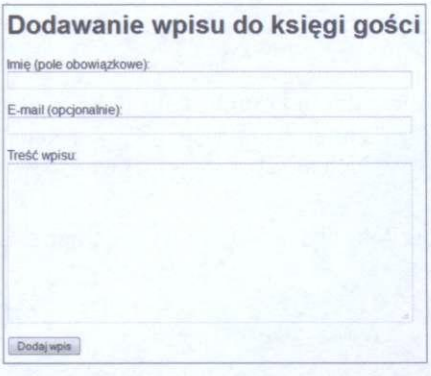
Dodaj kilka przykładowych wpisów do księgi gości, używając odpowiednio zbudowanej instrukcji `INSERT`, według wzoru podanego w przykładzie 5. Sposób wprowadzania poleceń SQL za pomocą narzędzia phpMyAdmin został opisany w przykładzie 4.

2.3. Tworzenie formularza dodającego wpisy do bazy danych

Utworzymy skrypt obsługujący dodawanie wpisów do księgi gości. Skrypt będzie wyświetlał formularz umożliwiający wprowadzenie danych. Dane z formularza będą wysyłane do tego samego skryptu. Jeżeli skrypt otrzyma dane z formularza, doda wpis do bazy danych.

Ćwiczenie 6.

Zaprojektuj w języku HTML formularz dodawania wpisów do księgi gości. Umieść w nim pola służące do wpisywania pojedynczej linii tekstu o nazwach *imie* i *email*, a także pole typu *textarea* o nazwie *wpis* (pamiętaj, że wielkość liter w nazwach pól ma znaczenie).



Rys. 3. *Przykładowy formularz dodawania wpisu do księgi gości*

Przykład 6. Dodawanie wpisów do księgi gości

Poniższy fragment skryptu umożliwia wyświetlanie formularza dodawania wpisów do księgi gości, obsługuje odbieranie danych i dodawanie wpisów. Skrypt powinien być zapisany w pliku o nazwie *dodaj-wpis.php*.

```
<h1>Dodawanie wpisu do księgi gości</h1>
<?php
// dołączenie kodu tworzącego połączenie z bazą danych
require('baza-danych.php');
$wyswietlFormularz = true;
if(isset($_POST['dodaj'])) {
  // odczytanie danych z formularza
  $imie = $_POST['imie'];
  $email = $_POST['email'];
  $wpis = $_POST['wpis'];
  $znalezionoBledy = false;
  // w tym miejscu powinien znaleźć się kod weryfikujący poprawność
  // wprowadzonych danych i ustawiający zmienną $znalezionoBledy na
  // true w przypadku znalezienia błędów
  if(!$znalezionoBledy) {
    $statement = $DB->prepare('INSERT INTO ksiega_gosci_wpisy (Imie, Email,
Wpis, Czas_dodania) VALUES(:imie, :email' :wpis, NOW())');
    $statement->execute(array(':imie' => $imie, ':email' => $email, ':wpis' =>
$wpis));
    echo('<p>Dziękujemy za dodanie wpisu. <a href="ksiega.php">Przejdź dalej</a>,
aby zobaczyć dodany wpis.</p>');
    $wyswietlFormularz = false;
  }
}
if($wyswietlFormularz) {
  ?>
  <form method="post" action="dodaj-wpis.php">
    <p>Imię (pole obowiązkowe): <br> <input type="text" name="imie" value=""></p>
    <p>E-mail (opcjonalnie): <br> <input type="text" name="email" value=""></p>
    <p>Treść wpisu: <br> <textarea name="wpis"></textarea></p>
    <p><input type="submit" name="dodaj" value="Dodaj wpis"></p>
  </form>
  <?php
}
?>
```

Warunek `if(isset($_POST['dodaj']))` sprawdza, czy przesłano dane z formularza. Zapis `$imie = $_POST['imie']` oznacza, że zmiennej *$imie* zostanie przypisana wartość pola *imie* pobrana z formularza (temat 31.). W ten sposób odczytujemy także wartości pozostałych pól formularza.

Następnie używamy pobranych wartości pól do zbudowania zapytania `INSERT`. Zapytanie budowane jest w dwóch krokach:

```
$statement = $DB->prepare('INSERT INTO ksiega_gosci_wpisy (Imie, Email,
Wpis, Czas_dodania) VALUES(:imie, :email, :wpis, NOW())');
```

```
$statement->execute(array(':imie' => $imie, ':email' => $email, ':wpis' =>
$wpis));
```

Najpierw tworzony jest szablon zapytania do wykonania, z miejscami na parametry (fragmenty po dwukropku, na przykład *:imie*, *:email*). Następnie zapytanie jest wykonywane przez metodę `execute`, do której przekazujemy tablicę z listą wartości dla nazwanych wcześniej parametrów. W rezultacie dane pobrane z formularza zostaną wstawione poleceniem `INSERT` do bazy danych. Dla pola *Czas_dodania* wstawiony zostanie natomiast aktualny czas.

Jeśli wszystko przebiegnie poprawnie, zostanie wyświetlony odpowiedni komunikat. W komunikacie został uwzględniony odnośnik do pliku *ksiega.php* – w dalszej części tematu pokażemy, jak stworzyć skrypt wyświetlający listę wpisów w księdze gości.

Ćwiczenie 7.

Rozbuduj skrypt z przykładu 6. tak, aby przed dodaniem wpisu sprawdzana była poprawność danych. Jeśli podczas sprawdzania zostaną wykryte błędy, powinien zostać wyświetlony odpowiedni komunikat i następnie ponownie formularz.
Aby po komunikacie o błędach skrypt ponownie wyświetlił formularz, należy ustawić wartość zmiennej *$znalezionoBledy* na wartość *true*.

Wskazówka: Aby sprawdzić, czy pole *imie* przesłane z formularza nie jest puste, możesz użyć konstrukcji `if(empty(trim($imie))) { ... }`. Aby sprawdzić, czy długość napisu nie przekracza maksymalnej dopuszczalnej długości dla danego pola, możesz użyć konstrukcji `if(mb_strlen($email) > 100) { ... }`.

2.4. Odczytywanie danych z bazy za pomocą instrukcji `SELECT`

Do pobierania listy wpisów z bazy danych użyjemy instrukcji `SELECT`:

```
SELECT * FROM ksiega_gosci_wpisy ORDER BY Czas_dodania DESC;
```

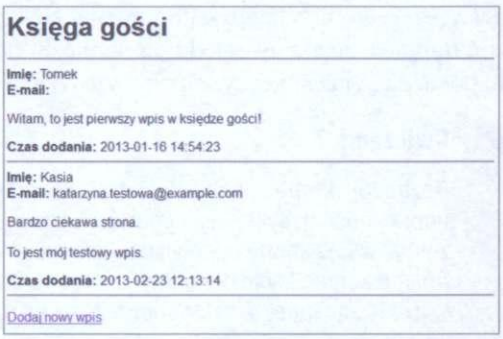
Zauważ, że polecamy bazie danych posortować listę wpisów od najnowszych do najstarszych.

Przykład 7. Fragment skryptu wyświetlającego listę wpisów z księgi gości

Kod skryptu utworzonego na bazie niniejszego przykładu powinien zostać zapisany w pliku o nazwie *ksiega.php*.

```
<h1>Księga gości</h1>
<hr />
<?php
// dołączenie kodu tworzącego połączenie z bazą danych
require('baza-danych.php');
foreach($DB->query('SELECT * FROM ksiega_gosci_wpisy') as $row) {
  echo('<b>Imię:</b> '.htmlspecialchars($row['Imie']).'<br>');
  echo('<b>E-mail:</b> '.htmlspecialchars($row['Email']).'<br>');
  echo('<p>'.nl2br(htmlspecialchars($row['Wpis'])).'</p>');
  echo('<b>Czas dodania:</b> '.$row['Czas_dodania'].'<br>');
  echo('<hr>');
}
?>
<p><a href="dodaj-wpis.php">Dodaj nowy wpis</a></p>
```

Dane wpisów z bazy danych odczytywane są w pętli `foreach`. Dla każdego pobranego rekordu generujemy kod HTML z danymi wpisu. Wartości pól pochodzące od użytkownika przetwarzamy przed wyświetleniem wbudowaną w PHP funkcją `htmlspecialchars()` – w przeciwnym wypadku użytkownik mógłby dodać do strony własne znaczniki HTML, które mogłyby popsuć jej wygląd. Tekst wpisu przetwarzamy dodatkowo funkcją `nl2br()`, która zamienia wprowadzone w formularzu znaki przejścia do nowej linii na znaczniki `<br>`, aby zostały one poprawnie wyświetlone przez przeglądarkę.



***Rys. 4.** Fragment strony prezentującej listę wpisów w księdze gości*

Ćwiczenie 8.

Przeanalizuj kod skryptu z przykładu 7. Zmień go tak, aby adres e-mail był wyświetlany w postaci hiperłącza (o ile wartość pola *Email* pobrana z bazy danych nie jest pusta).

Wskazówka: Kod HTML tworzący odnośnik do adresu e-mail ma postać:

`<a href="mailto:adres@email">adres@email</a>`).

Warto zapamiętać

- Biblioteka PDO znacznie upraszcza łączenie się z bazami danych z poziomu PHP. Udostępnia ona obiektowy interfejs, umożliwiający wykonywanie zapytań i przetwarzanie ich wyników.
- Kod odpowiadający za łączenie się z bazą danych i zawierający wymagane parametry najczęściej zapisuje się w osobnym pliku.
- Aby tworzyć nowe bazy danych i tabele na serwerze MySQL, można skorzystać z narzędzia phpMyAdmin. W domyślnej instalacji XAMPP narzędzie to dostępne jest pod adresem *http://localhost/phpmyadmin/*.
- Do wykonywania prostych zapytań, niezawierających parametrów pochodzących od użytkownika, możemy wykorzystać metodę `$DB->query(...)` wywoływaną dla utworzonego obiektu bazy danych. Kolejne rekordy z wyników zapytania możemy odczytywać za pomocą pętli `foreach`.
- Do wykonywania zapytań z parametrami zalecamy stosowanie metod `$statement = $DB->prepare(...)` i `$statement->execute()`. Gwarantują one, że parametry zostaną poprawnie umieszczone w treści zapytania SQL, nawet jeśli zawierają znaki specjalne (na przykład apostrofy czy cudzysłowy).

Pytania, problemy

1. Co to jest MySQL?
2. Czy baza danych musi być umieszczona na tym samym komputerze, co oprogramowanie serwera WWW?
3. W jaki sposób tworzymy konta użytkowników MySQL i bazy danych za pomocą narzędzia phpMyAdmin?
4. Dlaczego skrypt PHP tworzący połączenie z bazą danych najczęściej zapisuje się w osobnym pliku?
5. Do czego służą polecenia `SELECT` i `INSERT` w języku SQL?
6. Do czego służy pętla `foreach` w PHP?
7. Do czego służy księga gości na stronie internetowej? Z jakich elementów składa się prosta księga?
8. Do czego służy funkcja `htmlspecialchars()`?

Zadania

1. Zbierz utworzone w tym temacie skrypty i dopracuj je w formie kompletnego projektu księgi gości gotowej do wykorzystania na stronie WWW.
2. Dodaj do utworzonej księgi gości arkusze stylów CSS, które uatrakcyjnią jej wygląd.
3. Zmodyfikuj skrypt wyświetlający wpisy w księdze gości tak, aby wiersz z adresem e-mail był wyświetlany tylko wtedy, kiedy wartość pola *Email* zapisana w bazie danych nie jest pusta.
4. Rozbuduj skrypt dodający wpisy do księgi gości o sprawdzanie poprawności składni adresów e-mail. Sprawdzanie powinno następować tylko wtedy, gdy użytkownik wypełni pole z adresem e-mail – dopuszczamy możliwość pozostawienia tego pola pustego.

   Wskazówka: W PHP od wersji 5.2 dostępna jest funkcja `filter_var`, pozwalająca sprawdzać poprawność różnych typów danych, w tym adresów e-mail.
   Przykład użycia:

   ```
   <?php
   $email  = 'test@example.com';
   if(!filter_var($email, FILTER_VALIDATE_EMAIL)) {
     echo 'Adres e-mail jest niepoprawny!';
   }
   ?>.
   ```

5. W kodzie stworzonym w ćwiczeniu 7. w przypadku wystąpienia błędu ponownie wyświetlany jest formularz dodawania wpisu do księgi gości. Wyświetlone pola nie będą jednak zawierały danych wprowadzonych wcześniej przez użytkownika. Zmodyfikuj formularz, tak aby w takiej sytuacji zawierał wprowadzone wcześniej dane.

   Wskazówka: Kod wyświetlający pole *imie* możesz zmodyfikować w następujący sposób (całość powinna być umieszczona w jednej linii):

   ```
   <input type="text" name="imie" value="<?php echo empty($_
   POST['imie']) ? '' : htmlspecialchars($_POST['imie']); ?>">
   ```

   Dla pola *textarea* wartość umieszczana jest bezpośrednio pomiędzy znacznikami `<textarea>` i `</textarea>`, a nie w atrybucie `value=""`, tak jak dla pól *input*.

Dla zainteresowanych

6. Zmodyfikuj skrypt wyświetlający wpisy z księgi gości, tak aby w przypadku dużej liczby wpisów były one wyświetlane partiami – po 20 wpisów na stronie. Na dole strony powinny być wyświetlane odnośniki *Poprzednia strona* i *Następna strona*, umożliwiające przeglądanie kolejnych stron z wpisami.
   Wskazówka: Łączną liczbę wpisów w księdze możesz pobrać zapytaniem: `SELECT COUNT(*) FROM ksiega_gosci_wpisy;`.
   Aby polecenie `SELECT` zwracało tylko żądany zakres wpisów z księgi gości, możesz wykorzystać klauzulę `LIMIT` (jej opis znajdziesz w dokumentacji MySQL). Na przykład polecenie `SELECT * FROM ksiega_gosci_wpisy ORDER BY Czas_dodania DESC LIMIT 0,10;` zwraca 10 najnowszych wpisów (0 oznacza indeks pierwszego rekordu na liście wyników, licząc od zera, 10 oznacza liczbę rekordów do wyświetlenia).
7. Napisz prosty system umożliwiający publikowanie artykułów na stronie. Wykorzystaj kod stworzony wcześniej na potrzeby księgi gości. Dodaj możliwość komentowania artykułów przez użytkowników.
   Wskazówka: Komentarze przechowuj w osobnej tabeli, zawierającej pole z *Id* artykułu, do którego odnosi się komentarz. Dodaj klucz obcy dla tego pola, odwołujący się do tabeli z artykułami. (W MySQL klucze obce obsługiwane są dla tabel używających silnika InnoDB – informację, w jaki sposób zmienić typ tabeli i dodać klucz obcy, znajdziesz w dokumentacji MySQL.)
8. Przygotuj dodatkową wersję skryptu wyświetlającego listę wpisów w księdze gości, przeznaczoną dla administratora serwisu. Obok każdego wpisu powinien się pojawiać przycisk formularza umożliwiający usunięcie danego wpisu.
   Wskazówka: Przyciski dla poszczególnych wpisów umieść w ramach oddzielnych formularzy. Do usuwania rekordów z tabeli w bazie danych wykorzystaj instrukcję `DELETE FROM ksiega_gosci_wpisy WHERE Id_wpisu=<identyfikator>`, gdzie `<identyfikator>` to odczytany z formularza identyfikator wpisu do usunięcia.

Przeczytaj, jeśli chcesz wiedzieć więcej...

Bezpieczeństwo serwera

Opisana w tematach 31. i 32. konfiguracja XAMPP służy tylko do celów szkoleniowych. Tworząc publicznie dostępną witrynę, najlepiej skorzystać na początku z usług zewnętrznego dostawcy hostingu, który konfiguruje serwer i dba o aktualizację jego oprogramowania.