Lekcja I

Strona 1 - Przedstawiamy PHP i MySQL

Jeśli nie mieszkałeś na Marsie przez ostatnie sześć do ośmiu miesięcy, słyszałeś o oprogramowaniu open source (OSS). Ten ruch nabrał rozpędu, co zauważają nawet najwięksi chłopcy. Firmy, takie jak Oracle, Informix i wiele innych, wypuszczają swoje flagowe produkty bazodanowe dla tego dziecka-plakatu ruchu OSS, Linuksa. Posiadanie bardzo złożonego systemu RDBMS (relacyjnego systemu zarządzania bazami danych) jest dobe i dobrze, jeśli wiesz, co z tym zrobić. Ale może dopiero wchodzisz w świat baz danych. Przeczytałeś artykuł Jaya i chcesz umieścić własną stronę internetową opartą na danych. Ale okazuje się, że nie masz zasobów ani chęci posiadania serwera ASP lub drogiej bazy danych. Chcesz czegoś za darmo i chcesz, żeby działało z Uniksem. Wpisz PHP i MySQL. Te dwa elementy stanowią najlepszą kombinację dla witryn internetowych opartych na danych na naszej planecie. Połączenie zostało również nagrodzone Bazą Roku na Webcon98, gdzie otrzymało uroczą tiarę. MySQL to mały, kompaktowy serwer bazy danych, idealny do małych - i nie tak małych - aplikacji. Oprócz obsługi standardowego SQL (ANSI), kompiluje się na wielu platformach i ma wielowątkowość na serwerach Unix, co zapewnia doskonałą wydajność. Dla osób niebędących Unixami MySQL można uruchomić jako usługę w systemie Windows NT i jako normalny proces na komputerach z systemem Windows. PHP jest językiem skryptowym po stronie serwera. Jeśli widziałeś ASP, będziesz zaznajomiony z kodem osadzania na stronie HTML. Podobnie jak ASP, skrypt PHP jest przetwarzany przez serwer WWW. Gdy serwer bawi się kodem PHP, zwraca zwykły stary kod HTML do przeglądarki. Ten rodzaj interakcji pozwala na dość skomplikowane operacje. Oprócz bycia wolnym (MySQL ma pewne ograniczenia licencyjne), kombinacja PHP-MySQL jest również wieloplatformowa, co oznacza, że możesz rozwijać się w systemie Windows i służyć na platformie Unix. Ponadto PHP może być uruchamiany jako zewnętrzny proces CGI, samodzielny interpreter skryptów lub wbudowany moduł Apache.

Jeśli jesteś zainteresowany, PHP obsługuje również ogromną liczbę baz danych, w tym Informix, Oracle, Sybase, Solid i PostgreSQL - oraz wszechobecne ODBC. PHP obsługuje wiele innych funkcji, które znajdują się u progu rozwoju Internetu. Obejmują one uwierzytelnianie, XML, dynamiczne tworzenie obrazów, WDDX, obsługę pamięci współdzielonej i dynamiczne tworzenie dokumentów PDF, by wymienić tylko kilka. Jeśli to nie wystarczy, PHP jest łatwe do rozszerzenia, więc możesz zaprogramować swoje własne rozwiązanie, jeśli programujesz dobrze. Wreszcie, ponieważ oba wysiłki mają charakter współpracy, zawsze jest mnóstwo wsparcia z dokumentacji i list mailingowych. Błędy są naprawiane szybko, a żądania funkcji są zawsze słyszane, oceniane i, jeśli to możliwe, wdrażane. Dosyć gadać! Zobaczmy, co zamierzamy omówić w tym samouczku.

Strona 2 - Instalacja MySQL

Przejdźmy od razu, złapmy sobie kopię tych wspaniałych paczek i zacznijmy hakować! To nie jest proste. Istnieje wiele opcji dostępnych do uzyskania, kompilacji i instalacji oprogramowania. Zajmijmy się najpierw MySQL, ponieważ będziemy go potrzebować, zanim uruchomimy PHP.

Centrala MySQL to http://www.mysql.com/. Jak przystało na program jego wzrostu, na całym świecie znajduje się zillion luster, więc wyświadcz Internet za przysługę i wybierz ten, który jest najbliżej ciebie. W tym momencie masz mnóstwo możliwości. Jeśli jesteś osobą zorientowaną na zrób to, pobierz kod źródłowy. Jeśli nie jesteś tak odważny, istnieją pewne skompilowane pliki binarne dla innych platform, które są już dostępne do pobrania. Ponadto istnieje wersja shareware MySQL dla użytkowników Windows. Jest to starsza wersja MySQL. Jeśli chcesz najnowszą wersję, musisz kupić licencję. Istnieją również sterowniki ODBC, które pozwalają aplikacjom rozmawiać z MySQL. Różne inne ekscytujące fragmenty także czają się na stronie, więc spójrz. Wstępnie skompilowane wersje Uniksa i wersja

Windows są tak proste, jak rozpakowywanie i uruchamianie, i nie wymagają wiele wyjaśnień. Skompilujmy więc kod źródłowy. Użytkownicy Windows, pamiętaj, że musisz uruchomić mysqld w katalogu mysql / bin. Pobierz skompresowany plik do katalogu źródłowego i zdekompresuj go, a następnie rozpakuj za pomocą gzip i tar. Szybkim sposobem na to jest wpisanie:

gunzip <mysql-xxxx.tar.gz | tar xvf -

Xxxx to miejsce, w którym umieszczasz numer wersji. Spowoduje to utworzenie katalogu o nazwie mysql-xxxx, który zawiera wszystkie pliki źródłowe. Przejdź do tego katalogu, wpisując cd mysql-xxxx i sprawdź różne pliki README i INSTALL. To ratownicy w lepkich sytuacjach. MySQL zawiera wygodny skrypt konfiguracyjny. Po prostu wpisz ./configure i pozwól, aby rzeczy same o siebie zadbały. Jeśli chcesz określić, co się dzieje i gdzie, wpisz ./configure --help daje listę opcji do wyboru. Na przykład, jeśli kompilujesz na komputerze z małą ilością pamięci, możesz wybrać flagę --with-low-memory. Lubię MySQL, aby zainstalować w jednym poręcznym drzewie katalogów, a nie w różnych miejscach na moim komputerze, więc określam lokalizację instalacji za pomocą flagi --prefix. Możesz również określić wiele innych opcji, takich jak co skompilować i co pominąć. Załóżmy, że chcemy wszystkiego w / usr / local / mysql na naszym serwerze. Oznacza to, że wpiszemy ./configure

--prefix = / usr / local / mysql.

Skrypt konfiguracyjny uruchomi i sprawdzi twój system, a następnie zbuduje niezbędne pliki do pomyślnej kompilacji. Jeśli to się nie powiedzie, zazwyczaj otrzymasz pomocny komunikat o błędzie z informacją dlaczego. Dość często przekonasz się, że skrypt nie powiedzie się, gdy szuka bibliotek wątków. Sprawdź, czy masz zainstalowane na komputerze pthreads MIT, a jeśli nie, dodaj je. Użytkownicy Linuksa będą musieli pobrać LinuxThreads. Są to krytyczne biblioteki, które umożliwiają MySQL wielowątkowość (tzn. Uruchamianie wielu wersji samego siebie). Ostatnie dystrybucje Linuksa mogą już mieć zainstalowane te biblioteki. Jeśli wszystko pójdzie zgodnie z planem, po prostu wpisz make i idź po kawę. MySQL to złożony program i kompilacja zajmuje trochę czasu. Jeśli pojawi się błąd, sprawdź dokumentację, aby sprawdzić, czy w konkretnym systemie operacyjnym brakuje czegoś konkretnego. Następnie wpisz make install, a wszystkie niezbędne pliki zostaną zainstalowane we wszystkich punktach dodatkowych. Teraz jesteś prawie gotowy do rzutu! Jeśli jesteś dziewicą MySQL i nigdy wcześniej nie instalowałeś MySQL, musisz utworzyć domyślne uprawnienia, więc wpisz ... scripts / mysql_install_db, aby je ustawić. to jest to! Jesteśmy gotowi do rzucenia. Wszystko, co musimy zrobić, to dodać możliwość uruchamiania i zatrzymywania serwera podczas uruchamiania i zamykania systemu. I tak, jest na to również skrypt. Wpisanie startu mysql.server uruchamia serwer, a stop mysql.server zatrzymuje serwer. To naprawdę oczywiste. Aby ręcznie uruchomić serwer (aby móc grać bez ponownego uruchamiania), wprowadź katalog główny w instalacji MySQL (/ usr / local / mysql) i wpisz bin / safe_mysqld &. Jesteś w połowie drogi. Teraz w PHP.

Strona 3 - Instalacja PHP

Uff! Mam nadzieję, że już masz MySQL działające i działające. To była prawie zabawa! Teraz dla PHP ... Ten proces jest nieco łatwiejszy, ale szereg opcji jest olśniewający. Nie bądź jednak zniechęcony. Zawsze możesz wrócić później i ponownie skompilować PHP, aby dodać lub usunąć opcje w razie potrzeby.

Domem PHP jest http://www.php.net/. Witryna PHP jest kopalnią informacji, od list projektów po raporty o błędach. Podobnie jak w przypadku MySQL, powinieneś wybrać pobliskie lustro. Oczywiście będziesz chciał, aby sekcja Pobieranie pobierała PHP. Przeprowadzę cię przez instalację PHP3. Aby dowiedzieć się, jak radzić sobie z PHP4, przeczytaj szczegółowe instrukcje instalacji PHP4 w Webmonkey Julie. Twój wybór opcji jest nieco bardziej ograniczony. Dostępnych jest kilka

skompilowanych plików binarnych, ale są one eksperymentalne. Jeśli pracujesz na czymś innym niż platforma Windows, pobierz kod źródłowy i skompiluj go samodzielnie. Ale najpierw zajmijmy się Windows. Podczas korzystania z PHP powszechną praktyką jest tworzenie na komputerze z systemem Windows, a następnie uruchamianie witryny na serwerze Unix. Może się okazać, że zrobisz to sam, co oznacza, że musisz być biegły w instalacji na obu platformach. Chwyćmy plik binarny Windows i zdekompresujmy go za pomocą naszego ulubionego narzędzia dekompresyjnego Zip do katalogu na dysku C o nazwie php3. Dostarczony plik README omawia szczegółowo instalację, ale oto wersja Reader's Digest: Jeśli chcesz zainstalować PHP w folderze innym niż C: fp3, musisz edytować plik .inf dostarczany z PHP. W katalogu php3 znajdziesz wiele plików .dll. Weź wszystkie pliki .dll, które nie zaczynają się od php_ i przenieś je do katalogu Windows. Następnie zmień nazwę php.ini-dist na php3.ini i przenieś go do katalogu Windows. Jeśli otworzysz ten plik, zobaczysz wiele interesujących rzeczy do zmiany. Na razie „odkomentuj” linię:

extension = php3_mysql.dll

Jeśli używasz Apache for Win32, skonfiguruj Apache do rozpoznawania i analizowania plików PHP. W zależności od wersji Apache, której używasz, musisz dodać następujące informacje do pliku httpd.conf lub srm.conf:

ScriptAlias / php3 / "c: / path-to-php-dir /" AddType application / x-httpd-php3 .php3Action application / x-httpd-php3 "/php3/php.exe"

Lub jeśli używasz IIS lub PWS, kliknij prawym przyciskiem myszy php_iis_reg.inf i wybierz „Zainstaluj”. Musisz ponownie uruchomić IIS, aby zobaczyć tę zmianę. OK, teraz, gdy Windows jest już na uboczu, przejdźmy do Uniksa. Oczywiście będziemy kompilować z kodu źródłowego. Podobnie jak w przypadku MySQL, pobierz i rozpakuj kod źródłowy. Ponownie PHP zawiera skrypt konfiguracyjny. Jednak nie możesz uciec od tego, jak wybrać domyślne ustawienia. Uruchom ./configure -help | więcej, aby zobaczyć strony i strony nowych i interesujących opcji. Musisz zdecydować między kompilacją jako CGI lub jako moduł Apache. Jeśli używasz serwera Apache Web i jesteś w stanie go ponownie skompilować, użyj modułu: Jest szybszy i łatwiejszy w użyciu. W przeciwnym razie możesz przejść do wersji CGI. Musimy także skompilować obsługę MySQL.

Na razie zakładamy, że uruchamiamy moduł z obsługą MySQL. Jeśli chcesz dodać inne opcje lub inne biblioteki, możesz to zrobić później. Rodzaj:

./configure --with-apache = / path / do / apache / dir --with-mysql = / usr / local / mysql

Pomiń opcję -with-apache, jeśli tworzysz wersję CGI. Proces konfiguracji zostanie uruchomiony i wygeneruje odpowiednie pliki systemowe. Teraz po prostu wpisz make. Czas na kolejną kawę. Jeśli w tym momencie zaczniesz się denerwować i drżeć, nie przejmuj się tym. Wszyscy jesteśmy trochę niespokojni podczas naszej pierwszej instalacji PHP. Wypij trochę kawy. Jeśli utworzyłeś wersję CGI, jesteś teraz gotowy do rzutu. Po prostu skopiuj wynikowy plik wykonywalny do swojego pliku CGI. Dla użytkowników modułu Apache wpisz make install, aby skopiować pliki do katalogu Apache. Następnie postępuj zgodnie z instrukcjami, aby dodać moduł do Apache i ponownie skompilować. Musisz teraz powiedzieć swojemu serwerowi WWW, jak przetwarzać strony za pomocą programu PHP. Jeśli nie używasz Apache, musisz sprawdzić dokumentację serwera sieci Web, aby dowiedzieć się, jak przetworzyć dokumenty z rozszerzeniem .php3. Użytkownicy Apache 1.3.x mogą po prostu dodać aplikację AddType / x-httpd-php3 .php3 do pliku httpd.conf lub srm.conf. Jeśli używasz wersji CGI, musisz dodać następujące elementy przed AddType:

ScriptAlias / php3 / "/ path-to-php-dir /" AddType application / x-httpd-php3 .php3 Action application / x-httpd-php3 "/ php3 / php"

to jest to! Przy odrobinie szczęścia masz już uruchomiony MySQL i funkcjonowanie PHP. Nie zapomnij sprawdzić FAQ i dokumentacji, jeśli utkniesz. Wypróbuj także listy mailingowe. Teraz, kiedy już to wszystko udało się zrobić, pozwólmy, by ten ruch ruszył!

Strona 4 - Twój pierwszy skrypt

Będziesz zadowolony wiedząc, że naprawdę trudne rzeczy są za tobą. Instalacja oprogramowania jest zawsze czarną dziurą, ponieważ tak wiele zmian z systemu na system. Ale przy odrobinie szczęścia twoja baza danych działa i PHP jest kompilowane i instalowane z naszym serwerem sieciowym i może rozpoznawać dokumenty z rozszerzeniami .php3. Zanurkujmy i napiszmy nasz pierwszy skrypt. Utwórz plik tekstowy zawierający następujące elementy:

```
<html>

 <body>

<? php

$myvar = "Hello World";

echo $ myvar;

?>

</body>

</html>
```

Teraz wywołaj adres URL, na przykład http: //myserver/test.php3. Powinieneś zobaczyć stronę zawierającą tekst „Hello World”. Jeśli pojawi się komunikat o błędzie, sprawdź dokumentację PHP, aby sprawdzić, czy wszystko zostało poprawnie skonfigurowane. To jest to! To twój pierwszy skrypt PHP. Jeśli zobaczysz źródło HTML strony, zobaczysz, że jest tylko tekst. Witaj świecie

Dzieje się tak, ponieważ silnik PHP sprawdził stronę, przetworzył wszystkie znalezione bloki kodu i zwrócił tylko HTML. Pierwszą rzeczą, którą zauważysz w powyższym skrypcie, są ograniczniki. To są linie, które rozpoczynają <? Php. Oznacza to początek bloku kodu PHP, a?> Wskazuje koniec bloku. Siła PHP polega na tym, że można je umieścić w dowolnym miejscu - a mam na myśli wszędzie - w kodzie na wiele sposobów. Później zobaczymy kilka interesujących zastosowań, ale na razie nie przejmujmy się. Jeśli chcesz, możesz również skonfigurować PHP do używania krótkich tagów, <?, i?>, Ale nie są one zgodne z XML, więc bądź ostrożny. Jeśli dokonujesz przełączenia z ASP, możesz nawet skonfigurować PHP do używania ograniczników <% i%>.

Inną rzeczą, którą zauważysz, jest średnik na końcu każdej linii. Są one znane jako separatory i służą do odróżnienia jednego zestawu instrukcji od drugiego. Możliwe jest napisanie całego skryptu PHP w jednej linii i oddzielenie części średnikami. Ale to byłby bałagan, więc dodamy nową linię po każdym średniku. Pamiętaj tylko, że każda linia musi kończyć się średnikiem. Wreszcie widzisz, że słowo myvar zaczyna się od znaku dolara. Ten symbol mówi PHP, że jest to zmienna. Przypisaliśmy słowa „Hello World” do zmiennej $ myvar. Zmienna może również zawierać liczby lub tablicę. Tak czy inaczej, wszystkie zmienne zaczynają się od symbolu znaku dolara. Prawdziwa moc PHP pochodzi z jego funkcji. Są to w zasadzie instrukcje przetwarzania. Jeśli dodasz wszystkie opcjonalne dodatki do PHP, dostępnych jest ponad 700 funkcji. Więc możesz sporo zrobić. Dodajmy teraz do obrazu trochę MySQL

Strona 5 - Załaduj bazę danych

Teraz jesteśmy gotowi do podłączenia MySQL. Jeden przydatny sposób na sprawdzenie, jakie opcje są dostępne w PHP i co dzieje się na twoim serwerze, to użycie funkcji phpinfo (). Utwórz skrypt z następującymi elementami:

<html>

<body>

<? php

phpinfo();

?>

</body>

</html>

Zapisz i wyświetl ten skrypt za pośrednictwem serwera sieci Web. Zobaczysz stronę zawierającą przydatne i interesujące informacje, takie jak ta. Ta informacja mówi wszystko o twoim serwerze, wewnętrznych zmiennych środowiskowych serwera WWW, opcjach, które są kompilowane i włączane i wyłączane. W pierwszej sekcji Rozszerzenia poszukaj linii zaczynającej się od MySQL. Jeśli tego brakuje, z jakiegoś powodu MySQL nie przeszedł do PHP. Wróć i przejrzyj kroki instalacji i sprawdź dokumentację PHP, aby zobaczyć, czy coś przeoczyłeś. Jeśli jest tam MySQL, jesteś gotowy do uruchomienia. Zanim będziemy mogli uzyskać dane z MySQL, musimy umieścić w nim dane. na tym etapie nie jest to łatwy sposób. Większość skryptów PHP zawiera plik zrzutu, który zawiera wszystkie dane wymagane do utworzenia i wypełnienia bazy danych MySQL. Wady tego procesu są naprawdę poza zakresem tego poradnika, więc zrobię to za ciebie. MySQL używa własnej tabeli uprawnień użytkownika. Podczas konfiguracji automatycznie tworzony jest domyślny użytkownik (root) bez hasła. Dodanie innych użytkowników z różnymi uprawnieniami zależy od administratora bazy danych, ale mógłbym napisać cały inny artykuł na ten temat, więc będziemy trzymać się użytkownika root. Jeśli skonfigurujesz własny serwer i bazę danych, ważne jest, aby przypisać hasło do użytkownika root. W każdym razie, przejdźmy do bazy danych. Dla użytkowników Win32 przepraszam, ale wymaga to trochę pracy DOS. Musisz użyć okna DOS lub wpisać wszystko w oknie Uruchom. Nie zapomnij wpisać ścieżki do lokalizacji katalogu MySQL / bin za pomocą poleceń. Użytkownicy Uniksa mogą pracować z katalogu bin MySQL, ale może być konieczne uruchomienie każdego polecenia z ./, aby programy działały. Pierwszą rzeczą, którą musimy zrobić, jest utworzenie rzeczywistej bazy danych. W wierszu polecenia wpisz:

mysqladmin -u root tworzy mydb

Tworzy to bazę danych o nazwie „mydb". Flaga informuje MySQL, że robimy to jako użytkownik root. Następnie dodamy trochę danych za pomocą ulubionego przykładu wszystkich pracowników, bazy danych pracowników. Potrzebujemy tego pliku zrzutu, o którym wspomniałem wcześniej. Jeśli interesuje Cię, jak to wszystko działa, przejrzyj podręcznik dołączony do MySQL lub sprawdź http://www.turbolift.com/mysql/.

Skopiuj i wklej następujący tekst do pliku i zapisz go w katalogu bin MySQL. (Wywołam plik mydb.dump.)

CREATE TABLE employees (id tinyint(4) DEFAULT '0' NOT NULL AUTO_INCREMENT, first varchar(20), last varchar(20), address varchar(255), position varchar(50), PRIMARY KEY (id), UNIQUE id (id));INSERT

INTO employees VALUES (1,'Bob','Smith','128 Here St, Cityname','Marketing Manager');INSERT INTO employees VALUES (2,'John','Roberts','45 There St , Townville','Telephonist');INSERT INTO employees VALUES (3,'Brad','Johnson','1/34 Nowhere Blvd, Snowston','Doorman');

Jeśli linie się zawijają, upewnij się, że każda instrukcja wstawiania znajduje się w nowej linii. Teraz wstawimy go do bazy danych mydb. W wierszu polecenia wpisz:

mysql -u root mydb <mydb.dump

Nie powinieneś popełniać błędów. Jeśli to zrobisz, sprawdź nieprawidłowe zawijanie linii.

Strona 6 - Wyciągnij to

OK, teraz mamy nasze dane w bazie danych. Zróbmy coś z tym. Skopiuj i wklej poniższy plik do pliku tekstowego i zapisz go w drzewie dokumentów serwera sieci Web z rozszerzeniem .php3.

```
<html>

<body>

<?

php

$db = mysql_connect ("localhost", "root");

mysql_select_db ("mydb", $ db);

$result = mysql_query ("SELECT * FROM workers", $ db);

printf ("Imię:% s <br> n", mysql_result ($ wynik, 0, "pierwszy"));

printf ("Nazwisko:% s <br> n", mysql_result ($ wynik, 0, " last "));

printf (" Adres:% s <br> n ", mysql_result ($ wynik, 0," adres "));

printf (" Pozycja:% s <br> n ", mysql_result ($ wynik , 0, „position”));

?>

</body>

</html>
```

Wyjaśnijmy, co się tutaj dzieje. Funkcja mysql_connect () otwiera łącze do serwera MySQL na podanym hoście (w tym przypadku jest to localhost) wraz z nazwą użytkownika (root). Jeśli musisz podać hasło, również je tutaj dodasz. Wynik połączenia jest przechowywany w zmiennej $ db. mysql_select_db () następnie informuje PHP, że wszelkie zapytania są przeciwko bazie danych mydb. Możemy utworzyć wiele połączeń z bazami danych na różnych serwerach. Ale na razie zostawmy to. Następnie mysql_query () wykonuje całą ciężką pracę. Używając identyfikatora połączenia z bazą danych, wysyła linię SQL do serwera MySQL, który ma zostać przetworzony. Zwracane wyniki są przechowywane w zmiennej $ wynik. Wreszcie mysql_result () służy do wyświetlania wartości pól z naszego zapytania. Używając $ wynik, przechodzimy do pierwszego wiersza, który jest ponumerowany 0, i wyświetlamy wartość określonych pól. Składnia funkcji printf może wydawać się trochę dziwna, jeśli wcześniej nie korzystałeś z Perla lub C. W każdym z powyższych wierszy% s wskazuje, że zmienna w drugiej połowie

wyrażenia (np. Mysql_result ($ result, 0, „position”)) powinna być traktowana jako łańcuch i wydrukowana. Więcej informacji na temat printf można znaleźć w dokumentacji PHP. Więc mamy to. Z powodzeniem zastosowaliśmy, zainstalowaliśmy i skonfigurowaliśmy MySQL i PHP, a my wykonaliśmy prosty skrypt, aby pobrać pewne informacje. Później zrobimy kilka sprytnych rzeczy, aby wyświetlić wiele rekordów, a nawet wysłać dane do i z bazy danych.

Lekcja II

Strona 1 - Uzyskiwanie Loopy

W tej lekcji zamierzamy zanurkować bezpośrednio i utworzyć kilka prostych, ale przydatnych stron przy użyciu PHP i MySQL. Zacznijmy od wyświetlenia bazy danych, którą stworzyliśmy wczoraj, ale z nieco większym rozmachem. Najpierw zapytajmy naszą bazę danych za pomocą następujących kod.

```
<html>

<body>

<? php

$db = mysql_connect ("localhost", "root");

mysql_select_db ("mydb", $ db);

$result = mysql_query ("SELECT * FROM workers", $ db);

echo "<table border = 1> ";

echo" <tr> <td> Nazwa </td> <td> Pozycja </tr> n ";

while ($ myrow = mysql_fetch_row ($ result)) {

printf (" <tr> <td>% s % s </td> <td>% s </td> </tr> n ", $ myrow [1], $ myrow [2], $ myrow [3]);}

echo" </table> n ”;

?>

</body>

</html>
```

Prawdopodobnie zauważyłeś, że wprowadziliśmy tutaj kilka nowych funkcji. Najbardziej oczywista jest pętla while (). Pętla mówi, że dopóki są nowe wiersze danych do pobrania (za pomocą funkcji mysql_fetch_row ()), przypisz ten wiersz do zmiennej $ myrow. Następnie wykonaj instrukcje między nawiasami klamrowymi ({}). Spójrz na sekundę, a to powinno mieć sens. Funkcja mysql_fetch_row () wygląda bliżej. Mały problem z mysql_fetch_row () polega na tym, że zwraca tablicę, która obsługuje tylko odniesienia numeryczne do poszczególnych pól. Więc pierwsze pole jest określane jako 0, drugie jako 1 i tak dalej. W przypadku złożonych zapytań może to stać się koszmarem. Przyjrzyjmy się teraz bliżej pętli. Pierwsze kilka linii rozpoznasz z przykładu w lekcji 1. Następnie w pętli while () pobieramy wiersz z wyniku i przypisujemy go do tablicy $ myrow. Następnie drukujemy zawartość tablicy na ekranie za pomocą funkcji printf. Po tym pętli ponownie się pętli, a kolejny wiersz jest przypisany do $ myrow. Zrobi to, aż zabraknie rzędów do pobrania. Wspaniałą rzeczą w pętli while () jest to, że jeśli zapytanie nie zwraca żadnych rekordów, nie pojawi się komunikat o błędzie. Po raz pierwszy nie będzie żadnych danych do przypisania do $ myrow, a program po prostu przejdzie dalej. Ale jeśli zapytanie

nie zwróci żadnych danych, nie będziemy mogli poinformować użytkownika, a prawdopodobnie powinniśmy przekazać jakąś wiadomość. To jest możliwe, więc zróbmy to.

Strona 2 - Bądź informowany

Spójrz na ten skrypt

```
<html>

<body>

<?php

$db = mysql_connect("localhost", "root");

mysql_select_db("mydb",$db);

$result = mysql_query("SELECT * FROM employees",$db);

if ($myrow = mysql_fetch_array($result)) {

echo "<table border=1>\n";

echo "<tr><td>Name</td><td>Position</td></tr>\n";

do {

printf("<tr><td>%s %s</td><td>%s</tr>\n", $myrow["first"], $myrow["last"], $myrow["address"]);

} while ($myrow = mysql_fetch_array($result));

echo "</table>\n";

} else {

echo "Sorry, no records were found!";

}

?>

</body>

</html>
```

Wprowadzono tutaj wiele nowych funkcji, ale są one dość proste. Po pierwsze, istnieje funkcja mysql_fetch_array (). Jest to dokładnie to samo, co mysql_fetch_row () z jednym fajnym wyjątkiem: używając tej funkcji, możemy odnosić się do pól według ich nazw (takich jak $ myrow ["first"]) zamiast ich liczb. To powinno zaoszczędzić nam trochę bólu głowy. Wprowadziliśmy również pętlę do / while i instrukcję if-else. Instrukcja if-else mówi, że jeśli możemy przypisać wiersz do $ myrow, kontynuuj; w przeciwnym razie przejdź do sekcji else i zrób to, co tam jest. Pętla do / while jest odmianą pętli while () używanej na ostatniej stronie. Potrzebujemy pętli do / while z bardzo dobrego powodu: Z początkową instrukcją if przypisaliśmy pierwszy wiersz zwrócony przez zapytanie do zmiennej $ myrow. Jeśli w tym momencie wykonaliśmy zwykłą instrukcję while (taką jak while ($ myrow = mysql_fetch_row ($ wynik)), wykopalibyśmy pierwszy rekord ze zmiennej i zastąpiliby go drugim rekordem. pętla pozwala nam przetestować warunek po jednorazowym uruchomieniu kodu, więc nie ma szans, abyśmy przypadkowo pominęli wiersz, a na koniec, jeśli w ogóle nie zostaną zwrócone żadne rekordy, instrukcje zawarte w części else {} zostaną wykonane. Zobacz tę część w akcji, zmień instrukcję

SQL na SELECT * FROM pracowników WHERE id = 6 lub coś innego, co nie zwróci żadnych rekordów. Teraz rozszerzmy ten kod pętli i if-else, aby utworzyć jedną fantazyjną stronę.

Strona 3 - Łącz się inteligentnie

Weźmiemy tę zapętloną moc, której się właśnie nauczyliśmy, i użyjemy jej w bardziej praktycznym przykładzie. Zanim jednak przejdziemy tutaj, powinieneś wiedzieć, jak pracować z formularzami, querystringiem i metodami GET i POST. Jay zakrył to nie tak dawno temu, więc spójrz na jego artykuł, jeśli nie znasz go. Teraz zamierzam pracować z querystringiem. Jak powinieneś wiedzieć, istnieją trzy sposoby uzyskania informacji na temat kwerendy. Pierwszym jest użycie metody GET w formularzu. Drugim jest wpisanie informacji do adresu URL w przeglądarce. Po trzecie, możesz osadzić querystring w standardowym łączu. Wystarczy, że tag kotwicy będzie wyglądał mniej więcej tak:

<a href="http://my_machine/mypage.php3?id=1">.

Wykorzystamy tę technikę już teraz. Po pierwsze, ponownie wyślij zapytanie do naszej bazy danych i wymień nazwiska pracowników. Spójrz na poniższy skrypt. Wiele z tego powinno już wyglądać dość znajomo.

```
<html>

<body>

<? php

$ db = mysql_connect („localhost”, „root”); mysql_select_db („mydb”, $ db);

$ result = mysql_query („SELECT * FROM workers”, $ db); if ($ myrow = mysql_fetch_array ($ wynik))

 {

robić

 {

printf ("<a href=\t%s?id=%s">% s% s </a> <br> n ", $ PHP_SELF, $ myrow [" id "], $ myrow [" najpierw "], $ myrow [" last "]);

} while ($ myrow = mysql_fetch_array ($ wynik));

} else {

echo „Przepraszamy, nie znaleziono żadnych rekordów!”;

}

?>

</body>

</html>
```

Wszystko jest takie samo z wyjątkiem funkcji printf, więc spójrzmy na to szczegółowo. Najpierw zauważ, że każdy znak cudzysłowu poprzedzony jest odwrotnym ukośnikiem. Ukośnik odwrotny mówi PHP, aby wyświetlał następujący po nim znak, zamiast traktować go jako część kodu. Zwróć także uwagę na użycie zmiennej $ PHP_SELF. Ta zmienna, która przechowuje nazwę i lokalizację skryptu, jest

przekazywana wraz z każdą stroną PHP. Jest to pomocne tutaj, ponieważ chcemy, aby ten plik nazywał się sam. Używając $ PHP_SELF, możemy być pewni, że tak się stanie, nawet jeśli plik zostanie przeniesiony do innego katalogu - lub nawet innego komputera. Jak już wspomniałem, linki te przywołują stronę. Jednak po raz drugi do kwerendy zostaną dodane pewne informacje. PHP robi fajną rzecz, gdy widzi parę nazwa = wartość w kwerendie. Automatycznie tworzy zmienną o nazwie i wartości wskazanej przez kwerendę. Ta funkcja pozwala nam przetestować, czy jest to pierwszy czy drugi raz na tej stronie. Wszystko, co musimy zrobić, to zapytać PHP, czy istnieje zmienna $ id. Kiedy już poznam odpowiedź na to pytanie, za drugim razem mogę wyświetlić inny zestaw informacji. Oto jak:

```
<html>

<body>

<? php

$ db = mysql_connect („localhost”, „root”); mysql_select_db („mydb”, $ db);

// wyświetl indywidualny rekord

if ($ id) {

 $ result = mysql_query („SELECT * FROM workers WHERE id = $ id”, $ db);

$ myrow = mysql_fetch_array ($ wynik);

printf ("Imię:% s n <br>", $ myrow ["first"]);

printf ("Nazwisko:% s n <br>", $ myrow ["last"]);

printf ("Adres:% s n <br>", $ myrow ["adres"]);

printf ("Pozycja:% s n <br>", $ myrow ["pozycja"]);

} else {

 // pokaż listę pracowników $ result = mysql_query („SELECT * FROM workers”, $ db);

if ($ myrow = mysql_fetch_array ($ wynik))

 {

// wyświetl listę, jeśli istnieją rekordy do wyświetlenia

zrobić {

printf ("<a href=\t%s?id=%s">% s% s </a> <br> n ", $ PHP_SELF, $ myrow [" id "], $ myrow [" najpierw "], $ myrow [" last "]);

} while ($ myrow = mysql_fetch_array ($ wynik));

} else {

// brak rekordów do wyświetlenia echa „Przepraszamy, nie znaleziono żadnych rekordów!”;

 }

}

?>
```

</body>

</html>

Ten kod jest teraz skomplikowany, więc zacząłem używać komentarzy, aby śledzić, co się dzieje. Możesz użyć //, aby utworzyć komentarz jednowierszowy lub / * i * /, aby rozpocząć i zakończyć duży blok komentarza. I mamy to: twój pierwszy naprawdę użyteczny skrypt PHP / MySQL! Przyjrzyjmy się teraz, jak podłączyć do niego formularze i wysłać informacje z powrotem do bazy danych.

Strona 4 - Wrzuć kilka formularzy

Udało nam się uzyskać dane z bazy danych bez większych trudności. Ale co z wysyłaniem danych w drugą stronę? To nie problem z PHP. Najpierw stwórzmy stronę z prostym formularzem.

<html>

<body>

<form method = "post" action = "<? php echo $ PHP_SELF?>">

Imię: <input type = "Text" name = "first">

Nazwisko: <wejście type = "Text" name = "last">

Adres: <input type = "Text" name = "address">

Położenie: <input type = "Text" name = "position">

<input type = "Submit" name = "submit" value = "Wprowadź informacje"> </form> </body> </html>

Zwróć uwagę na ponowne użycie $ PHP_SELF. Tak jak powiedziałem w lekcji 1, możesz używać PHP w dowolnym miejscu kodu HTML. Zauważysz również, że każdy element formularza odpowiada nazwie pola w bazie danych. To nie jest obowiązkowe; to po prostu dobry pomysł, abyś mógł później objąć swój kod. Zauważ również, że nadałem przyciskowi Submit atrybut name. Zrobiłem to, aby móc sprawdzić, czy istnieje zmienna $ submit. W ten sposób, gdy strona zostanie ponownie wywołana, będę wiedział, czy ktoś użył tego formularza. Powinienem wspomnieć, że nie musisz mieć strony, która sama się pętli. Jeśli chcesz, możesz objąć dwie, trzy lub więcej stron. W ten sposób wszystko pozostaje zwarte.

OK, dodajmy kod, który sprawdzi wejście formularza. Aby udowodnić, że dane wejściowe formularza przechodzą, zrzucę wszystkie zmienne na ekran za pomocą $ HTTP_POST_VARS. Jest to przydatna funkcja debugowania. Jeśli kiedykolwiek będziesz musiał zobaczyć wszystkie zmienne na stronie,

use $GLOBALS.

<html>

<body>

<?php

if ($submit) {

// process form

while (list($name, $value) = each($HTTP_POST_VARS))

{ echo "$name = $value
\n";

```
}
} else{
// display form
?>
<form method="post" action="<?php echo $PHP_SELF?>">
First name:<input type="Text" name="first"><br>
Last name:<input type="Text" name="last"><br>
Address:<input type="Text" name="address"><br>
Position:<input type="Text" name="position"><br>
<input type="Submit" name="submit" value="Enter information">
</form>
<?php}
// end
 if
?>
</body>
</html>
```

Teraz, gdy wygląda to dobrze, weźmy informacje o formularzu i opublikujmy je w bazie danych.

```
<html>
<body>
<?php
if ($submit) {
// process form
$db = mysql_connect("localhost", "root");
mysql_select_db("mydb",$db);
$sql = "INSERT INTO employees (first,last,address,position) VALUES ('$first','$last','$address','$position')";
 $result = mysql_query($sql); echo "Thank you! Information entered.\n";
} else {
// display form
?>
```

<form method="post" action="<?php echo $PHP_SELF?>">

First name:<input type="Text" name="first">

Last name:<input type="Text" name="last">

Address:<input type="Text" name="address">

Position:<input type="Text" name="position">

<input type="Submit" name="submit" value="Enter information">

</form>

<? php

} // end if

?>

</body>

</html>

Wprowadziłeś już dane do bazy danych. Wciąż jest daleki od doskonałości. Co zrobić, jeśli ktoś pozostawia puste pole lub wprowadza tekst, gdy chcemy wprowadzić numer? Co, jeśli gdzieś jest błąd? Nie martw się. Dotrzemy do tego.

Strona 5 - Uczyń formularze bardziej inteligentnymi

W tym samouczku ładowałem instrukcję SQL do zmiennej ($sql) przed uruchomieniem zapytania w bazie danych za pomocą mysql_query (). Jest to przydatne do debugowania. Jeśli coś pójdzie nie tak, zawsze możesz powtórzyć SQL na ekranie, aby sprawdzić, czy nie ma błędów. Wiemy już, jak pobrać dane do bazy danych. Spróbujmy teraz zmodyfikować rekordy, które są już w bazie danych. Edycja danych łączy dwa elementy, które już widzieliśmy: wyświetlanie danych na ekranie i wysyłanie danych z powrotem do bazy danych za pomocą formularza. Jednak edycja jest nieco inna, ponieważ musimy pokazać odpowiednie dane w formularzu. Najpierw przetworzmy kod z lekcji 1, aby wyświetlić nazwy pracowników na naszej stronie. Ale tym razem wypełnimy nasz formularz informacjami o pracownikach. Powinno to wyglądać trochę tak:

<html>

<body>

<?php

$db = mysql_connect("localhost", "root");

mysql_select_db("mydb",$db);

if ($id) { // query the DB $sql = "SELECT * FROM employees WHERE id=$id"; $result = mysql_query($sql);

$myrow = mysql_fetch_array($result);

?>

<form method="post" action="<?php echo $PHP_SELF?>">

```
<input type=hidden name="id" value="<?php echo $myrow["id"] ?>">

 First name:<input type="Text" name="first" value="<?php echo $myrow["first"] ?>"><br>

Last name:<input type="Text" name="last" value="<?php echo $myrow["last"] ?>"><br>

Address:<input type="Text" name="address" value="<?php echo $myrow["address"] ?>"><br>

Position:<input type="Text" name="position" value="<?php echo $myrow["position"] ?>"><br>

<input type="Submit" name="submit" value="Enter information">

</form>

 <?php

} else {

// display list of employees

$result = mysql_query("SELECT * FROM employees",$db); while ($myrow = mysql_fetch_array($result))

{

printf("<a href=\"%s?id=%s\">%s %s</a><br>\n", $PHP_SELF, $myrow["id"], $myrow["first"], $myrow["last"]);

}

}

?>

</body>

</html
```

Po prostu powtórzyliśmy informacje o polu w atrybucie wartości każdego elementu, co było dość łatwe. Zbudujmy na tym trochę więcej. Dodamy możliwość wysyłania edytowanego kodu z powrotem do bazy danych. Ponownie użyjemy przycisku Prześlij, aby przetestować, czy musimy przetwarzać dane wejściowe formularza. Zwróć także uwagę na nieco inną instrukcję SQL, której używamy.

```
<html>

<body>

<?php

$db = mysql_connect("localhost", "root");

mysql_select_db("mydb",$db);

if ($id) { if ($submit) { $sql = "UPDATE employees SET first='$first',last='$last',address='$address',position='$position' WHERE id=$id";

$result = mysql_query($sql);

echo "Thank you! Information updated.\n";
```

```
} else {

// query the DB

$sql = "SELECT * FROM employees WHERE id=$id"; $result = mysql_query($sql); $myrow = mysql_fetch_array($result);

?>

 <form method="post" action="<?php echo $PHP_SELF?>">

<input type=hidden name="id" value="<?php echo $myrow["id"] ?>">

 First name:<input type="Text" name="first" value="<?php echo $myrow["first"] ?>"><br>

Last name:<input type="Text" name="last" value="<?php echo $myrow["last"] ?>"><br> Address:<input type="Text" name="address" value="<?php echo $myrow["address"] ?>"><br> Position:<input type="Text" name="position" value="<?php echo $myrow["position"] ?>"><br>

<input type="Submit" name="submit" value="Enter information">

</form>

<?php

}

} else {

// display list of employees

 $result = mysql_query("SELECT * FROM employees",$db); while ($myrow = mysql_fetch_array($result))

{

printf("<a href=\"%s?id=%s\">%s %s</a><br>\n", $PHP_SELF, $myrow["id"], $myrow["first"], $myrow["last"]);

}

}

?>

</body>

</html>
```

I to jest to. Udało nam się połączyć większość funkcji, które widzieliśmy, w jeden skrypt. Możesz również zobaczyć, jak użyliśmy instrukcji if () w innej instrukcji if (), aby sprawdzić wiele warunków. Nadszedł czas, aby to wszystko połączyć i stworzyć jeden skrypt zabójcy.

Strona 6 - Wszystko razem teraz

Skończymy tę lekcję, umieszczając wszystko na jednej stronie, która może dodawać, edytować i usuwać wpisy z bazy danych. To rozszerzenie tego, co do tej pory omówiliśmy i stanowi dobrą recenzję. Spójrzmy.

```
<html>

<body>

<?php

$db = mysql_connect("localhost", "root");mysql_select_db("mydb",$db);if ($submit)

{

// here if no ID then adding else we're editing

 if ($id) { $sql = "UPDATE employees SET first='$first',last='$last',address='$address',position='$position' WHERE id=$id";

 } else {

$sql = "INSERT INTO employees (first,last,address,position) VALUES ('$first','$last','$address','$position')";

} // run SQL against the DB

$result = mysql_query($sql); echo "Record updated/edited!<p>";} elseif ($delete)

{

// delete a record

$sql = "DELETE FROM employees WHERE id=$id"; $result = mysql_query($sql);

echo "$sql Record deleted!<p>";

} else {

// this part happens if we don't press submit

 if (!$id) {

// print the list if there is not editing

$result = mysql_query("SELECT * FROM employees",$db); while ($myrow = mysql_fetch_array($result))

{ printf("<a href=\"%s?id=%s\">%s %s</a> \n", $PHP_SELF, $myrow["id"], $myrow["first"], $myrow["last"]);

printf("<a href=\"%s?id=%s&delete=yes\">(DELETE)</a><br>", $PHP_SELF, $myrow["id"]);

}

}

?>

<P> <a href="<?php echo $PHP_SELF?>">ADD A RECORD</a>

<P> <form method="post" action="<?php echo $PHP_SELF?>">

<?php
```

```
if ($id)
{
// editing so select a record
 $sql = "SELECT * FROM employees WHERE id=$id";
$result = mysql_query($sql);
$myrow = mysql_fetch_array($result);
$id = $myrow["id"];
$first = $myrow["first"];
$last = $myrow["last"];
$address = $myrow["address"];
 $position = $myrow["position"];
// print the id for editing
?>
<input type=hidden name="id" value="<?php echo $id ?>">
<?php
}
?>
First name:<input type="Text" name="first" value="<?php echo $first ?>"><br>
 Last name:<input type="Text" name="last" value="<?php echo $last ?>"><br>
Address:<input type="Text" name="address" value="<?php echo $address ?>"><br>
Position:<input type="Text" name="position" value="<?php echo $position ?>"><br>
<input type="Submit" name="submit" value="Enter information"> </form>
<? php
}
?>
</body>
</html>
```

Wygląda to skomplikowanie, ale tak naprawdę nie jest. Skrypt jest podzielony na trzy części. Pierwsza instrukcja if () sprawdza, czy przycisk Prześlij został naciśnięty, a jeśli tak, sprawdza, czy istnieje zmienna $ id. Jeśli nie, dodajemy rekord. W przeciwnym razie edytujemy rekord. Następnie sprawdzamy, czy istnieje zmienna $ delete. Jeśli tak, usuwamy rekord. Zauważ, że z pierwszą instrukcją if ()

sprawdziliśmy zmienną, która przeszła jako POST, iw tej zmiennej zmienna byłaby częścią GET. Wreszcie podejmujemy domyślną akcję, która wyświetla listę pracowników i formularz. Ponownie sprawdzamy, czy istnieje zmienna $ id. Jeśli istnieje, wysyłamy zapytanie do bazy danych, aby wyświetlić odpowiedni rekord. W przeciwnym razie wyświetlamy pusty formularz. Włożyliśmy wszystko, czego się nauczyliśmy, w jeden skrypt. Użyliśmy pętli while () i instrukcji if (), a my wykonaliśmy gamę podstawowych instrukcji SQL - SELECT, INSERT, UPDATE i DELETE. Wreszcie przyjrzeliśmy się, w jaki sposób możemy przekazywać informacje z jednej strony do drugiej za pomocą adresów URL i wprowadzania formularzy.

Lekcja III

Strona 1 - Miejsce na wszystko

Witamy w trzeciej i ostatniej lekcji tego samouczka. Jeśli przeszedłeś przez lekcję 1 i lekcję 2, znasz już podstawy do instalowania i pisania przydatnych skryptów z MySQL i PHP. Przyjrzymy się kilku użytecznym funkcjom PHP, które powinny znacznie ułatwić Ci życie. Po pierwsze, spójrzmy na pliki dołączone. Wszyscy znamy podstawy dołączania, prawda? Do zawartości pliku zewnętrznego odwołuje się i importuje do pliku głównego. To całkiem proste: dzwonisz do pliku i jest on dołączony. Kiedy robimy to w PHP, musimy porozmawiać o dwóch funkcjach: include () i require (). Różnica między tymi dwiema funkcjami jest subtelna, ale ważna, więc przyjrzyjmy się bliżej. Funkcja require () działa w sposób podobny do XSSI; pliki są dołączane jako część oryginalnego dokumentu, gdy tylko ten plik zostanie przeanalizowany, niezależnie od jego lokalizacji w skrypcie. Jeśli więc zdecydujesz się umieścić funkcję require () w pętli warunkowej, zewnętrzny plik zostanie uwzględniony, nawet jeśli ta część pętli warunkowej jest fałszywa. Funkcja include () importuje plik przywoływany przy każdym napotkaniu. Jeśli nie zostanie napotkany, PHP nie będzie się tym przejmować. Oznacza to, że możesz użyć include w pętlach i instrukcji warunkowych, a one będą działać dokładnie tak, jak zaplanowano.

Wreszcie, jeśli użyjesz require (), a plik, który zawierasz, nie istnieje, twój skrypt zatrzyma się i wygeneruje błąd. Jeśli użyjesz include (), twój skrypt wygeneruje ostrzeżenie, ale kontynuuj. Możesz to sprawdzić samodzielnie, wykonując poniższy skrypt. Uruchom skrypt, a następnie zastąp include () poleceniem require () i porównaj wyniki.

```
<html>

<body>

<? php

include ("emptyfile.inc");

echo „Hello World”;

?>

</body>

</html>
```

Lubię używać przyrostka .inc z moimi plikami załączników, więc mogę je oddzielić od zwykłych skryptów PHP. Jeśli to zrobisz, upewnij się, że ustawiłeś plik konfiguracyjny serwera sieci Web w celu przeanalizowania plików .inc jako plików PHP. W przeciwnym razie hakerzy mogą odgadnąć nazwę

plików dołączanych i wyświetlić je w przeglądarce jako pliki tekstowe. Może to być złe, jeśli masz poufne informacje - takie jak hasła do bazy danych - zawarte w załącznikach. Co zatem zrobisz z plikami dołączanymi? Prosty! Umieść informacje wspólne dla wszystkich znajdujących się na nich stron. Rzeczy takie jak nagłówki HTML, stopki, kod połączenia z bazą danych i funkcje zdefiniowane przez użytkownika są dobrymi kandydatami. Wklej ten tekst do pliku o nazwie header.inc.

<? php

$ db = mysql_connect („localhost”, „root”); mysql_select_db („mydb”, $ db);

?>

<html>

<head>

<title>

<? php echo $ title?>

</title>

</head>

<body>

 <h2> <? php echo $ title?> </h2>

Następnie utwórz kolejny plik o nazwie footer.txt, który zawiera odpowiedni tekst zamykający i znaczniki. Teraz stwórzmy trzeci plik zawierający rzeczywisty skrypt PHP. Wypróbuj następujący kod, upewniając się, że serwer MySQL jest uruchomiony.

<? php

$ title = „Hello World”;

include („header.inc”);

$ result = mysql_query („SELECT * FROM workers”, $ db);

echo "<table border = 1> n";

echo "<tr> <td> Nazwa </td> <td> Pozycja </tr> n";

while ($ myrow = mysql_fetch_row ($ result)) {

printf ("<tr> <td>% s% s </td> <td>% s </tr> n", $ myrow [1], $ myrow [2], $ myrow [3]);}

}

echo „</table> n”;

include („footer.inc”);

?>

Zobacz co się dzieje? Pliki dołączane są wrzucane do głównego pliku, a następnie całość jest wykonywana przez PHP. Zwróć uwagę, jak zmienna $ title została zdefiniowana przed odwołaniem do header.inc. Jego wartość jest udostępniana kodowi w header.inc; stąd tytuł strony jest zmieniany.

Możesz teraz użyć header.inc na wszystkich stronach PHP, a wszystko, co musisz zrobić, to zmienić wartość $ title z strony na stronę. Korzystając z kombinacji dołączeń, HTML, instrukcji warunkowych i pętli, można tworzyć złożone odmiany do strony z absolutnym minimum kodu. Obejmuje to szczególnie przydatne w przypadku korzystania z funkcji, co zobaczymy w drodze. W ekscytujący świat walidacji danych.

Strona 2 - Prosta walidacja

Wyobraź sobie przez chwilę, że nasza baza danych jest ładnie rozplanowana i teraz zwracamy się do użytkowników o informacje, które zostaną wstawione do bazy danych. Co więcej, wyobraźmy sobie, że w bazie danych znajduje się pole oczekujące na wprowadzenie danych liczbowych, takich jak cena. Na koniec wyobraź sobie, że twoja aplikacja spada na krzyczącą stertę, ponieważ niektóre inteligentne alki umieszczają tekst w tym polu. MySQL nie chce widzieć tekstu w tej części instrukcji SQL - i narzeka gorzko.

Co robić? Czas na potwierdzenie. Sprawdzanie poprawności oznacza po prostu, że zbadamy fragment danych, zwykle z formularza HTML, i sprawdzimy, czy pasuje on do określonego modelu. Może to być od zapewnienia, że element nie jest pusty, do sprawdzenia, czy element spełnia określone kryteria (na przykład, czy określono wartość liczbową lub czy adres e-mail zawiera @ dla adresu e-mail). Sprawdzanie poprawności można wykonać po stronie serwera lub po stronie klienta. PHP jest używany do sprawdzania poprawności po stronie serwera, podczas gdy JavaScript lub inny język skryptowy oparty na kliencie może zapewnić walidację po stronie klienta. Ten artykuł dotyczy PHP, więc skupimy się na końcu serwera. Ale jeśli szukasz gotowych skryptów walidacyjnych po stronie klienta, sprawdź bibliotekę kodu Webmonkey. Zignorujmy na razie naszą bazę danych i skoncentrujmy się na walidacji PHP. Jeśli chcesz, możesz dodać dodatkowe pola do naszej bazy danych pracowników po prostu za pomocą instrukcji MySQL ALTER - to znaczy, jeśli chcesz zatwierdzić wartości, które sprawdzimy. Istnieje kilka przydatnych funkcji PHP, których możemy użyć do sprawdzenia poprawności naszych danych, a ich zakres waha się od prostych do bardzo złożonych. Prostą funkcją, której możemy użyć, może być strlen (), która mówi nam o długości zmiennej. Bardziej złożoną funkcją może być ereg (), która używa pełnej obsługi wyrażeń regularnych do złożonych zapytań. Nie będę tutaj zagłębiać się w zawiłości regexu, ponieważ napisano na ten temat całe książki, ale podam kilka przykładów na następnej stronie. Zacznijmy od prostego przykładu. Sprawdzimy, czy zmienna nie istnieje

```
<html>

<body>

<?php

if ($submit) {

if (!$first || !$last) { $error = "Sorry! You didn't fill in all the fields!";

}

else {

// process form

echo "Thank You!";

}

}
```

```
if (!$submit || $error)

{

echo $error;

?>

<P> <form method="post" action="<?php echo $PHP_SELF ?>">

 FIELD 1: <input type="text" name="first" value="<?php echo $first ?>"> <br>

FIELD 2: <input type="text" name="last" value="<?php echo $last ?>"> <br>

<input type="Submit" name="submit" value="Enter Information">

</form>

<?php

} // end if

?>

</body>

</html>
```

Kluczami do tego skryptu są zagnieżdżone instrukcje warunkowe. Pierwsze sprawdza, czy przycisk Prześlij został naciśnięty. Jeśli tak, sprawdza, czy obie zmienne $ first i $ last istnieją. The || symbol oznacza „lub” i! symbol oznacza „nie”. Moglibyśmy również przepisać instrukcję, aby powiedzieć: „Jeśli $ najpierw nie istnieje lub $ last nie istnieje, to ustaw błąd $ na następujący”. Następnie rozszerzmy trochę rzeczy, sprawdzając, czy łańcuch ma określoną długość. Byłoby to idealne rozwiązanie dla haseł, ponieważ nie chcesz, aby jakiś leniwy użytkownik wprowadzał hasło zawierające tylko jedną lub dwie litery. Wolałbyś, żeby to było, powiedzmy, sześć lub więcej postaci. Funkcją tego jest, jak już wiesz, strlen (). Zwraca po prostu liczbę równą liczbie znaków w testowanej zmiennej. Tutaj zmodyfikowałem powyższy skrypt, aby sprawdzić długość $ first i $ last.

```
<html>

<body>

<?php

if ($submit) {

if (strlen($first) < 6 || strlen($last) < 6) { $error = "Sorry! You didn't fill in all the fields!";

} else {

// process form

echo "Thank You!";

}

}

if (!$submit || $error) {
```

```
 echo $error;

?>

<P> <form method="post" action="<?php echo $PHP_SELF ?>">

 FIELD 1: <input type="text" name="first" value="<?php echo $first ?>"> <br>

FIELD 2: <input type="text" name="last" value="<?php echo $last ?>"> <br>

<input type="Submit" name="submit" value="Enter Information">

</form>

 <?php

} // end if

?>

</body>

</html>
```

Uruchom ten skrypt i spróbuj wpisać sześć lub mniej liter, aby zobaczyć, co się stanie. To proste, ale całkiem skuteczne.

Strona 3 - Niezbyt prosta weryfikacja

Porozmawiajmy trochę o używaniu wyrażeń regularnych z funkcjami ereg() i eregi(). Jak powiedziałem wcześniej, mogą być one dość skomplikowane lub bardzo proste, w zależności od potrzeb. Używając wyrażeń regularnych, możesz zbadać łańcuch i inteligentnie wyszukać wzory i odmiany, aby sprawdzić, czy pasują one do ustawionych kryteriów. Najpopularniejszy z nich polega na sprawdzeniu, czy adres e-mail jest prawidłowy (chociaż oczywiście nie ma bezpiecznego sposobu na wykonanie tego). Zamiast zagłębiać się w tajemnice wyrażeń regularnych, podam kilka przykładów. Możesz użyć tego samego formularza, który utworzyliśmy na poprzedniej stronie - wystarczy wkleić poniższe wiersze, aby zobaczyć, jak działają.

Po pierwsze, upewnijmy się, że tylko tekst został wprowadzony do elementu formularza. To wyrażenie regularne sprawdza, czy użytkownik wprowadził jedno lub więcej małych liter, od a do z. Żadne numery nie są dozwolone:

```
if (! ereg ("[a-Z]", $ first) ||! ereg ("[a-Z]", $ last)) {
```

Teraz rozszerzmy to wyrażenie, aby sprawdzić, czy łańcuch ma długość od czterech do sześciu znaków. Użycie [[: alpha:]] jest łatwym sposobem sprawdzenia poprawności znaków alfabetycznych. Liczby w nawiasach klamrowych sprawdzają liczbę wystąpień. Zauważ, że ^ i $ wskazują początek i koniec ciągu.

```
if (! ereg ("^ [[: alpha:]] {4,6} $", $ pierwszy) ||! ereg ("^ [[: alpha:]] {4,6} $", $ last) )
```

Na koniec utwórzmy wyrażenie regularne, które sprawdzi poprawność adresu e-mail. Było wiele dyskusji na temat skuteczności sprawdzania adresów e-mail w ten sposób. Nic nie jest całkowicie niezawodne, ale to, co mam poniżej, działa całkiem dobrze. Zabrałem ten klejnot z listy mailingowej PHP. To świetny zasób - użyj go. I tak, wygląda to tak przerażająco, jak się wydaje.

```
if (! ereg ('^ [-! # $% &' * + ./ 0-9 =? AZ ^ _`az {|} ~] + '.
```

' @ '.

' [-! # $ % & '* + / 0-9 =? AZ ^ _`az {|} ~] +'. '[-! # $% &' * + ./ 0-9 =? AZ ^ _`az {|} ~] + $ ', $ last)) {

Nie spędzaj zbyt wiele czasu na oglądaniu tego.

Strona 4 - Funkcje

Ciesz się tym ostatnim wyrażeniem regex? Zabawnie, prawda? Czy nie byłoby jeszcze przyjemniej wejść w ten fragment na kilkunastu różnych stronach, które muszą przetwarzać adresy e-mail ?! Pomyśl o radości ze znalezienia literówki w tym bałaganie - i rób to kilkanaście razy mniej. Ale oczywiście jest lepszy sposób. Pamiętasz, kiedy rozmawialiśmy o plikach dołączanych wcześniej w tej lekcji? Pozwoli nam to na utworzenie fragmentu kodu, takiego jak sprawdzanie poczty e-mail, i włączenie go wielokrotnie na kilku stronach. W ten sposób, gdy chcemy zmienić kod, musimy edytować tylko jeden plik, nie wiele. Ale jeśli chcemy to zrobić, będziemy musieli użyć funkcji. Korzystaliśmy już z funkcji wiele razy. Za każdym razem, gdy sprawdzamy bazę danych lub sprawdzamy długość ciągu używamy funkcji. Funkcje te są wbudowane w PHP. Jeśli jesteś zapalonym koderem, możesz rozszerzyć PHP o własne, dostosowane funkcje. Ale to trochę zaawansowane w tym tutorialu. Zamiast tego stworzymy funkcje, które będą rezydować w naszym skrypcie PHP. Funkcja jest po prostu blokiem kodu, do którego przekazujemy jedną lub więcej wartości. Funkcja przetwarza następnie informacje i zwraca wartość. Funkcja może być tak prosta lub złożona, jak nam się podoba, ale tak długo, jak możemy przekazać wartość i wydostać ją, tak naprawdę nie obchodzi nas, jak jest złożona. To piękno funkcji. Funkcje w PHP zachowują się podobnie do funkcji w C. Kiedy definiujemy funkcje, musimy określić, jakich wartości funkcja może oczekiwać. Trudno sobie z tym poradzić, ale zapobiega to dziwnym wydarzeniom. Dzieje się tak, ponieważ zmienne wewnątrz funkcji są znane jako zmienne prywatne. Oznacza to, że istnieją tylko wewnątrz funkcji. Możesz na przykład mieć w skrypcie zmienną o nazwie $ myname. Jeśli utworzyłeś funkcję i spodziewałeś się użyć tej samej zmiennej $ myname (o tej samej wartości), to nie zadziałałoby. Alternatywnie, możesz mieć zmienną $ myname w swoim skrypcie, a także utworzyć inną zmienną o nazwie $ myname w swojej funkcji, a te dwie będą współistniały dość szczęśliwie z oddzielnymi wartościami. Nie polecam tego robić! Kiedy wrócisz i edytujesz go sześć miesięcy później, będziesz łamał rzeczy z lewej i prawej strony. Istnieją wyjątki od tej reguły, jak w przypadku wszystkich rzeczy, ale to wykracza poza zakres tego artykułu. Stwórzmy więc funkcję. Zaczniemy po prostu. Musimy nadać funkcji nazwę i powiedzieć, jakich zmiennych oczekiwać. Musimy również zdefiniować funkcję, zanim ją nazwiemy.

```
<html>

<body>

<?php

function addnum($first, $second)

{

$newnum = $first + $second;

}

echo addnum(4,5);

?>

</body>
```

```
</html>
```

To jest to! Najpierw stworzyliśmy naszą funkcję. Zwróć uwagę, jak zdefiniowaliśmy dwie nowe zmienne, zwane $ first i $ second. Kiedy wywołujemy funkcję, każdej zmiennej przypisywana jest wartość w oparciu o kolejność, w jakiej pojawia się na liście - 4 przechodzi do $ najpierw, 5 do $ sekund. Następnie po prostu dodaliśmy dwie liczby i zwróciliśmy wynik. „Powrót" oznacza po prostu wysłanie wyniku z powrotem. Na końcu skryptu drukujemy liczbę 9. Stwórzmy coś, co jest bardziej przydatne w naszej aplikacji bazodanowej. A może coś, co z wdziękiem obsługuje błędy? Spróbuj tego:

```
<html>

<body>

<? php

function do_error($error)

{

echo "Hmm, looks like there was a problem here...<br>";

echo "The reported error was $error.\n<br>";

echo "Best you get hold of the site admin and let her know."; die;

}

if (!$db = @mysql_connect("localhost","user", "password"))

{

$db_error = "Could not connect to MySQL Server"; do_error($db_error);}

?>

</body>

</html>
```

Przed uruchomieniem spróbuj zamknąć MySQL lub użyć fałszywej nazwy użytkownika lub hasła. Otrzymasz miły, użyteczny komunikat o błędzie. Uważni czytelnicy zauważą symbol @ przed mysql_connect (). To eliminuje komunikaty o błędach, dzięki czemu informacje są uzyskiwane tylko z funkcji. Zobaczysz również, że byliśmy w stanie przekazać zmienną do funkcji, która została zdefiniowana gdzie indziej. Pamiętaj, że powiedziałem, że funkcje używają własnych zmiennych prywatnych? To było trochę białe kłamstwo. W rzeczywistości można tworzyć zmienne poza funkcją dostępną dla funkcji. Możesz utworzyć funkcję do wysyłania zapytań do bazy danych i wyświetlania zestawu wyników na kilku stronach. Nie chcesz za każdym razem przekazywać identyfikatora połączenia z bazą danych do funkcji. W tej sytuacji możesz udostępnić kod połączenia jako zmienną globalną. Na przykład:

```
<html>

<body>

<? php

function db_query($sql)
```

```
{

 global $db;

$result = mysql_query($sql,$db);

return $result;}$sql = "SELECT * FROM mytable";$result = db_query($sql);

?>

</body>

</html>
```

Jest to podstawowa funkcja, ale chodzi o to, że nie trzeba wysyłać $ db podczas wywoływania funkcji - można ją udostępnić za pomocą słowa globalnego. W tej instrukcji możesz zdefiniować inne zmienne jako globalne, po prostu oddziel nazwy zmiennych przecinkiem. Wreszcie możesz wyglądać jak prawdziwy profesjonalista, używając opcjonalnych zmiennych funkcji. W tym przypadku kluczem jest zdefiniowanie zmiennej do pewnych wartości domyślnych w funkcji, a następnie wywołanie funkcji bez określenia wartości dla zmiennej spowoduje przyjęcie wartości domyślnej. Ale jeśli podasz wartość, będzie miała pierwszeństwo. Zmieszany? Na przykład, gdy łączysz się z bazą danych, prawie zawsze łączysz się z tym samym serwerem i prawdopodobnie użyjesz tej samej nazwy użytkownika i hasła. Ale czasami musisz połączyć się z inną bazą danych. Spójrzmy.

```
<html>

<body>

<?php

function db_connect($host = "localhost", $user="username", $pass="graeme")

{

$db = mysql_connect($host, $username, $password);

return $db;

}

$old_db = db_connect();

$new_host = "site.com";

$new_db = db_connect($new_host);

?>

</body>

</html>
```

Czy to nie fajne? Zmienne używane wewnątrz funkcji zostały zdefiniowane podczas definiowania funkcji. Przy pierwszym wywołaniu funkcji używane są wartości domyślne. Za drugim razem łączymy się z nowym hostem, ale z tą samą nazwą użytkownika i hasłem. Świetna sprawa! Zastanów się, gdzie możesz użyć innych funkcji w kodzie. Możesz ich użyć do sprawdzania danych, wykonywania rutynowych zadań i tak dalej. Używam ich bardzo często podczas przetwarzania tekstu do wyświetlenia

na stronie internetowej. Mogę sprawdzać, analizować i modyfikować tekst, aby dodać nowe linie i uciec za pomocą znaków HTML. Teraz pozostaje tylko przekazać kilka słów mądrości.

Strona 5 - Doradztwo końcowe

Jeśli chodzi o bazy danych, trzeba się wiele nauczyć. Jeśli jeszcze tego nie zrobiłeś, znajdź dobrą książkę na temat projektowania baz danych i naucz się tworzyć solidną bazę danych - na dowolnej platformie. To nieoceniona umiejętność, która pozwoli Ci zaoszczędzić mnóstwo czasu i bólu głowy na dłuższą metę. Dowiedz się także o MySQL. Jest to skomplikowana, ale interesująca baza danych z dużą ilością przydatnej dokumentacji. Dowiedz się o strukturze tabeli, typach danych i SQL. Możesz uzyskać naprawdę imponujące rzeczy, jeśli znasz wystarczająco dużo SQL. Wreszcie jest PHP. Witryna PHP ma prawie wszystko, czego potrzebujesz, od obszernego podręcznika po archiwa list mailingowych do repozytoriów kodów. Doskonałym sposobem na poznanie PHP jest zapoznanie się z przykładami użytymi w podręczniku i sprawdzenie archiwów kodu. Wiele opublikowanych skryptów składa się z funkcji lub klas, których można używać za darmo we własnych skryptach bez konieczności wymyślania koła. Ponadto lista mailingowa jest doskonałym miejscem do sprawdzenia, jeśli utkniesz. Twórcy sami czytają listę i jest tam wielu dobrze poinformowanych ludzi, którzy mogą ci pomóc w drodze.

Powodzenia i dobrego kodowania!