Школа

PHP

Данная серия документов подготовлена на основе материалов сайта Школы Консорциума W3C. Этот сайт является экспериментальным сервером, на котором содержание документов хранится в формате XML. Пользователям сайта эти документы доступны в виде HTML (преобразование на строне клиента с помощью таблицы стилей XSLT) и в виде PDF (преобразование тех же документов в XSL-FO, а затем в формат PDF).

Добро пожаловать в школы PHP и PHP/MySQL

Школа PHP[1] (open link)

В школе PHP вы узнаете что из себя представляет язык PHP. PHP - это гипертекстовый препроцессор и язык скриптов, исполняемых на стороне сервера. Изучайте PHP!

Школа PHP/MySQL[2] (open link)

Для многих главной причиной изучения PHP являются возможности, которые он открывает при работе с базами данных. На этой школе вы узнаете как можно использовать PHP и базу данных MySQL для хранения информации и размещения ее на веб-сайте.

Ресурсы по PHP[3] (open link)

Список полезных ресурсов по PHP в Интернете.

Школа PHP: содержание

1: http://xml.nsu.ru/php/php_intro.xml
2: http://xml.nsu.ru/php/php2_intro.xml
3: http://xml.nsu.ru/php/php_resources.xml
4: http://xml.nsu.ru/php/php_intro.xml
5: http://xml.nsu.ru/php/php_part2.xml
6: http://xml.nsu.ru/php/php_part3.xml
7: http://xml.nsu.ru/php/php_part4.xml
8: http://xml.nsu.ru/php/php_part5.xml
9: http://xml.nsu.ru/php/php_part6.xml

Заключение[10] (open link)

Несколько дополнительных замечаний.

Школа PHP/MySQL: содержание

Введение[11] (open link)

Здесь излагаются основы совместного использования PHP и MySQL.

Создание базы данных[12] (open link)

Как создать базу данных в MySQL и подготовить ее к приему данных.

Вставка информации[13] (open link)

Как используя PHP установить полный контакт с базой данных и как добавлять информацию в новую базу данных.

Отображение данных[14] (open link)

Как для вставки информации использовать переменные и формы и как отображать информацию из базы данных.

Другие способы вывода[15] (open link)

Как форматировать выходную информацию и как отбирать различную информацию из базы данных.

Отдельные записи и обработка ошибок[16] (open link)

Еще несколько способов работы с получаемой из базы данных информации, как контролировать ошибки подсчетом рядов.

Обновление и удаление[17] (open link)

Как создать страницу для обновления и удаления информации из базы данных.

Завершение скрипта[18] (open link)

Как собрать написанный вами скрипт воедино и несколько дополнительных замечаний о работе с базой данных MySQL.

Введение в PHP

Предисловие

10: http://xml.nsu.ru/php/php_part7.xml
11: http://xml.nsu.ru/php/php2_intro.xml
12: http://xml.nsu.ru/php/php2_part2.xml
13: http://xml.nsu.ru/php/php2_part3.xml
14: http://xml.nsu.ru/php/php2_part4.xml
15: http://xml.nsu.ru/php/php2_part5.xml
16: http://xml.nsu.ru/php/php2_part6.xml
17: http://xml.nsu.ru/php/php2_part7.xml
18: http://xml.nsu.ru/php/php2_part8.xml

До последнего времени в Интернете мало кто даже пытался использовать скриптовые языки, не говоря уже об их серьезном применении. Но сейчас все больше и больше людей создают свои веб-сайты и скриптовые языки приобретают важное значение. В связи с этим скриптовые языки становятся все более простыми для изучения, при этом язык PHP оказывается одним из самых простых и мощных скриптовых языков.

Что такое PHP?

PHP представляет собой пре-процессор гипертекста (Hypertext Preprocessor) и исполняется на стороне сервера. Он выполняется на вашем веб-сервере, а не в браузере пользователя, так что вам не приходится думать о проблемах совместимости. PHP - сравнительно новый язык (по сравнению с Perl (CGI) и Java), но быстро становится одним из самых популярных скриптовых языков в Интернете.

Почему именно PHP?

Может быть, вы хотите знать, почему среди скриптовых языков следует выбирать PHP, а не, например, Perl, или даже зачем вам вообще знать скриптовые языки. Сначала о скриптовых языках вообще. Изучение скриптового языка, или даже только понимание его, откроет для вашего сайта огромные новые возможности. Хотя вы можете скачать готовые скрипты с сайтов, подобных Hotscripts (www.hotscripts.com), они часто содержат рекламу своего автора или делают не совсем то, что вам надо. Имея понимание скриптового языка, вы легко можете отредактировать эти скрипты, чтобы они делали именно то, что надо, или даже написать свои собственные.

Использование скриптов на вашем сайте позволит добавить на него множество новых "интерактивных" элементов, типа форм обратной связи, гостевых книг, форумов, счетчиков, и даже более сложных вещей, вроде портальной системы, системы управления содержанием и рекламой и т.д. Используя такие вещи на своем сайте, вы обнаружите, что он выглядит более профессионально. Кроме того, всякому, кто работает в индустрии веб-разработки, гораздо легче найти хорошую работу, если знать скриптовые языки.

Что мне понадобится?

Как уже говорилось, PHP - это скриптовый язык, исполняемый на стороне сервера. Это означает, что хотя пользователям не понадобится устанавливать у себя какое-то новое программное обеспечение, ваш веб-хост должен иметь на сервере установленный PHP. Вы можете узнать об этом у специалистов, поддерживающих сервер, но о наличии на сервере PHP можно узнать и с помощью самого первого скрипта на нашей школе. Если ваш сервер не поддерживает PHP, вы мождете попросить его установить, благо, что PHP можно бесплатно скачать. Если вам нужен недорогой хостинг с поддержкой PHP, можно порекомендовать HostRocket.

Написание PHP

Писать PHP-скрипт очень легко. Вам для этого не понадобится никакого специального программного обеспечения, кроме обычного текстового редакотора (например, стандартного Блокнота). Запустите его и вы готовы написать свой первый PHP-скрипт.

Объявление PHP

PHP-скрипты всегда заключены в пару PHP-тэгов. Они сообщают серверу, что заключенная в тэги информация должна обрабатываться как PHP. Это можно сделать тремя способами:

```
<?
Здесь располагается PHP-код
?>

<?php
Здесь располагается PHP-код
php?>

<script language="php">
Здесь располагается PHP-код
</script>
```

Все эти записи имеют одно и то же значение, но на этой школе мы будем пользоваться первым способом записи (<? и ?>). Для этого нет какой-то особой причины, так что вы можете пользоваться любым способом объявления PHP. Однако, нужно помнить, что ваш код должен начинаться и заканчиваться одним и тем же тэгом (например, нельзя начать тэгом <?, а закончить тэгом </script>).

Ваш первый скрипт

Первый PHP-скрипт, который вы напишете, будет очень простым. Все, что он делает - выводит информацию о PHP, установленном на вашем сервере. Напишите в текстовом редакторе следующий код:

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

Как вы видите, здесь всего лишь одна строчка кода. Это стандартная функция PHP, которая называется phpinfo. Он дает серверу указание вывести стандартную таблицу, содержащую информацию об установленном на сервере PHP.

Вы, наверное, обратили внимание, что строка заканчивается точкой с запятой. Это очень важный момент. Как и во многих других языках программирования, почти все строки кода заканчиваются точкой с запятой, и если вы где-то забыли об этом, это вызовет ошибку.

Завершение скрипта и его тестирование

Теперь вы закончили написание скрипта. Сохраните его как phpinfo.php и выложите обычным способом на свой сервер. Теперь с помощью браузера откройте URL вашего документа. Если он работает (и если PHP установлен на сервере), вы получите большую страницу, на которой будет много информации о PHP на вашем сервере.

Если скрипт не работает и вы получили пустую страницу, вы либо допустили в коде ошибку, либо сервер не поддерживает эту функцию (хотя мне не приходилось встречать сервера, которые ее не поддерживают). Если вместо отображения страницы появился диалог, предлагающий скачать этот файл, значит PHP не установлен на вашем сервере и вам нужно либо

подыскать другой хостинг, либо попросить его установить.

Сохраните этот файл для дальнейшего использования.

Отображение информации и переменные

Вывод информации в браузере и хранение информации в переменных

Печать текста

Выводить текст с помощью PHP-скрипта очень просто. Как и большинство других вещей, PHP позволяет делать это несколькими различными способами. Тем не менее, основным способом является использование команды print. Эта команда позволяет выводить на экран текст, переменные или их комбинации.

Выражение print применяется следующим образом:

```
print("Hello world!");
```

Разберем эту строчку кода: print - это команда, которая указывает скрипту что делать. За ней идет информация, которая должна быть выведена, она заключена в скобки. Поскольку мы выводим текст, он дополнительно заключен в кавычки. И, наконец, строка заканчивается точкой с запятой, как и почти каждая строка в PHP. А все вместе следует поместить в стандартные тэги PHP. В результате получится следующий код:

```
<?
print("Hello world!");
?>
```

Он выведет на экран текст:

```
Hello world!
```

Переменные

Как и в других языках программирования, в PHP можно задавать переменные. В PHP имеется несколько типов переменных, но чаще всего используются переменные типа String (строка). Переменые этого типа могут содержать текст и числа. Все строки начинаются со знака $. Чтобы приравнять строчную переменную какому-либо тексту, используется следующий код:

```
$welcome_text = "Привет, добро пожаловать на мой сайт.";
```

В этой строчке кода нет ничего сложного. Все, что находится внутри кавычек, станет содержимым переменной. Нужно только помнить несколько правил строчных переменных:

- Имена строчных переменных чувствительны к регистру: например, $Welcome_Text - это не та же самая переменная, что и $welcome_text
- Имена строчных переменных могут содержать числа и знак подчеркивания, но они не могут начинаться с чисел или знака подчеркивания

Когда вы приравниваете строчную переменную числу, его не обязательно заключать в кавычки:

```
$user_id = 987
```

Вывод переменных

Для отображения переменной на экране применяется такой же код, что и для отображения текста, хотя и немного видоизмененный. Следующий код выведет на экран содержимое переменной $welcome_text:

```
<?
$welcome_text = "Привет, добро пожаловать на мой сайт.";
print($welcome_text);
?>
```

Как вы видите, единственное отличие заключается в том, что в команде print имя переменной не нужно заключать в кавычки.

Форматирование текста

К сожалению, обычный внешний вид выводимой из PHP-программ информации, выглядит довольно скучным. Она выводится обычным шрифтом браузера, установленным в нем по умолчанию. Однако, используя HTML отформатировать текст очень легко. Это потому, что поскольку PHP исполняется на стороне сервера, в страницу можно внести изменения еще до того, как она отправляется пользователю. Это означает, что отправляется только результат работы скрипта, так что в выше приведенном примере браузеру будет отправлен только текст:

```
Привет, добро пожаловать на мой сайт.
```

Но ведь в скрипт и в переменные вполне можно включить стандартную HTML-разметку. Единственная проблема в том, что во многих HTML-тэгах применяется знак двойной кавычки " и может возникнуть путаница: кавычки, применяемые в HTML-тэгах перепутаются с кавычками, применяемыми в команде print. Нужно как-то сообщить скрипту, какие кавычки нужно рассматривать как часть PHP-команды print (они отмечают начало и конец выводимого текста), а какие нужно игнорировать (являющиеся частью HTML-кода).

В этом примере я хочу, чтобы текст выводился шрифтом Arial красного цвета. Обычный HTML-код для этого выглядит так:

```
<font face="Arial" color="#FF0000">
</font>
```

Как вы можете заметить, в этом коде имеется четыре кавычки, которые могут запутать скрипт. Для того, чтобы это не случилось, перед каждой кавычкой мы добавляем обратный слэш, который укажет скрипту, что данную кавычку следует игнорировать. Код будет теперь выглядеть так:

```
<font face=\"Arial\" color=\"#FF0000\">
</font>
```

Теперь можно включить эту разметку в команду print:

```
print("<font face=\"Arial\" color=\"#FF0000\">
Привет, добро пожаловать на мой сайт.</font>");
```

В результате в браузере пользователя наше приветствие будет выведено шрифтом Arial красного цвета, поскольку браузеру отправляется только такой код:

```
<font face="Arial" color="#FF0000">
Привет, добро пожаловать на мой сайт.</font>
```

Кажется, что все это затрудняет передачу браузеру обычного HTML-кода, но далее на нашей школе мы познакомимся с другим способом, который несколько проще.

Выражение IF

Основы работы с условными выражениями, позволяющими ветвить скрипт

Основы работы с выражением IF

Выражения IF применяются для сравнения двух значений и выполнения на основе проверки различных действий. Выражение IF состоит из трех частей: раздела IF, разделов THEN и ELSE. Раздел IF содержит проверяемое условие. Если условие выполняется, исполняется раздел THEN, если нет - исполняется раздел ELSE.

Структура выражения IF

Структура выражения IF выглядит так:

```
IF (что-то одно == что-то другое) {
Раздел THEN
} else {
Раздел ELSE
}
```

Переменные

Чаще всего выражение IF применяется для сравнения переменной с каким-либо текстом, числом или другой переменной, например:

```
if ($username == "webmaster")
```

Здесь значение переменной сравнивается с текстовой строкой. Раздел кода THEN будет выполняться только в том случае, если значение переменной в точности равно содержимому текстовой строки, заключенной в кавычки. Таким образом, если переменная имеет значение "Webmaster" или "WEBMASTER", условие выполнено не будет.

Конструирование раздела THEN

Теперь можно добавить в скрипт раздел THEN:

```
if ($username == "webmaster") {
echo "Пожалуйста, введите пароль";
}
```

Этот код выведет данный текст только в том случае, если имя пользователя (username) равно "webmaster". Если нет, то выводиться на экран ничего не будет. Вполне можно оставить выражение IF в таком виде, поскольку раздел ELSE не обязателен. Это особенно полезно в случае применения множественных выражений IF.

Конструирование раздела ELSE

Добавить в условное выражение раздел ELSE не сложнее, чем раздел THEN. Просто добавьте дополнительный код:

```
if ($username == "webmaster") {
echo "Пожалуйста, введите пароль";
} else {
echo "Извините, но вы - неизвестный пользователь";
}
```

Конечно, вы можете не ограничиваться только одной строчкой кода. Вы можете внутрь круглых скобок разделов THEN и ELSE вставлять любое количество команд PHP. Вы даже можете в них вставлять другие выражения IF (вложенные выражения).

Другие сравнения

Есть и другие способы применения выражения IF для сравнения различных значений. Во-первых, вы можете проверить, не равны ли значения двух переменных, например:

```
if ($enteredpass == $password)
```

Также вы можете использовать стандартные символы для проверки больше-меньше:

```
if ($age < "13")
```

или :

```
if ($date > $finished)
```

Кроме того внутри раздела IF можно проверять сразу несколько условий. Например, вы проверяете, чтобы все поля формы были заполнены. Для этого можно использовать такой код:

```
if ($name == "" || $email == "" || $password == "") {
echo "Пожалуйста, заполните все поля";
}
```

Циклы и множства

Цикл WHILE

Цикл WHILE - одна из самых полезных команд в PHP. Его очень легко организовывать и пользоваться. Цикл WHILE выполняет определенный кусок кода до тех пор, пока выполняется какое-либо условие.

Повторение куска кода заданное число раз

Если у вас есть кусок кода, который вы хотите выполнить несколько раз без того, чтобы его вписывать в скрипт несколько раз, вы используете цикл WHILE. Например, если вы хотите вывести приветствие "Hello World" 5 раз, вы можете использовать следующий код:

```
$times = 5;
$x = 0;
while ($x < $times) {
echo "Hello World";
++$x;
}
```

Разберемся в этом коде. В первых двух строчках устанавливаются переменные. Переменная

$times содержит число раз, сколько должно повториться приветствие. Переменная $x - это счетчик, в котором отмечается сколько уже раз было выведено приветствие. Далее идет строка WHILE. В ней дается указание повторять кусок кода, выводящий приветствие до тех пор, пока $x меньше, чем $times. Далее идет повторяемый кусок кода. Он заключен в фигурные скобки: { }.

После строки, содержащей команду echo (она выводит текст), идет еще очень важная строка:

```
++$x;
```

Это означает то же самое, что и запись:

```
$x = $x + 1;
```

В этой строке к значению $x прибавляется единица. Эта часть кода продолжает циклически выполняться, пока $x не станет равным 5 (нужно число повторений), после этого начинает выполняться дальнейший код.

Использование $x

Переменная, в которой фиксируется число проходов цикла ($x в данном примере) может использоваться не только как счетчик. Например, если вы хотите создать веб-страницу, на которой написаны все числа от 1 до 1000, вы можете выводить все эти числа по одному, но гораздо лучше использовать следующий код:

```
$number = 1000;
$current = 0;
while ($current < $number) {
++$current;
echo "$current<br>";
}
```

Отметим здесь несколько моментов. Во-первых, мы поместили выражение ++$current; до команды echo. Это для того, чтобы числа начали выводиться не с 0, а с 1. Строка изменения счетчика (у нас ++$current;) может быть размещена в любом месте цикла WHILE, ее расположение не имеет значения. И конечно, она может не только прибавлять, но и вычитать, умножать, делить - делать что угодно.

Еще одна причина того, что мы поместили выражение ++$current; до команды echo состоит в том, что иначе бы наш цикл остановился после выведения числа 999, потому что к этому моменту счетчик бы достиг 1000 и цикл бы закончился, потому что условие WHILE было бы выполнено.

Массивы

Массивы имеются во многих языках программирования. Это особые переменные, которые могут хранить не одно, а несколько значений - каждое хранится в своем пронумерованном "месте" массива. Массивы чрезвычайно полезны, особенно при совместном использовании с циклами WHILE.

Создание массива

Создание массива только слегка отличается от создания обыкновенной переменной. В этом

примере мы создаем массив, в котором хранятся пять имен:

```
$names[0] = 'John';
$names[1] = 'Paul';
$names[2] = 'Steven';
$names[3] = 'George';
$names[4] = 'David';
```

Как вы видите, все элементы массива пронумерованы, начиная с 0. Чтобы внести значение в элемент массива нужно указать его номер в квадратных скобках [ ].

Чтение из массива

Чтение из массива происходит также, как и запись в него. Для этого нужно только указать номер элемента массива, который вы хотите прочитать. Если, например, нужно прочитать значение третьего элемента массива, можно использовать следующий код:

```
echo "Третье имя: $names[2]";
```

Этот код выведет:

```
Третье имя: Steven
```

Применение массивов и циклов

Одно из лучших применений циклов - вывод информации из массива. Допустим, мы хотим вывести список имен:

```
Имя 1: John
Имя 2: Paul
Имя 3: Steven
Имя 4: George
Имя 5: David
```

Для этого можно использовать следующий код:

```
$number = 5
$x = 0
while ($x < $number) {
$namenumber = $x + 1
echo "Имя $namenumber: $names[$x]<br>";
++$x
}
```

Как вы видите, мы используем переменную цикла $x для указания на выводимые элементы массива. Кроме того, вы наверное заметили, что здесь используется переменная $namenumber, которая всегда на 1 больше, чем $x. Она нужна из-за того, что нумерация массива начинается с 0, а при выводе на печать нумерацию нужно начинать с единицы.

Работа с e-mail

Отправление e-mail с использованием PHP

Команда Mail

В PHP очень легко отправлять e-mail, в отличие от других скриптовых языков, где для этого может понадобиться специальная установка, (например, в CGI). Для отправки корреспонденции нужна всего лишь одна команда, mail(). Она применяется следующим образом:

```
mail($to,$subject,$body,$headers);
```

В этом примере в качестве параметров команды mail() используются переменные с понятными именами, но вместо переменных тут может использоваться и просто текст. Первый параметр $to. Он содержит адрес e-mail, на который высылается письмо. В парметре $subject указывается тема письма, а в параметре $body содержится сам текст письма.

Параметр $headers используется для любых дополнительных e-mail-заголовков, которые вы захотите использовать. Чаще всего он применяется для задания информации в поле "From" отправляемого e-mail, но также сюда можно включить информацию о полях "cc" и "bcc".

Отправка E-mail

Прежде, чем отправить письмо, если вы используете в качестве параметров переменные, вам нужно задать значение этих переменных. Вот пример кода, отправляющего электронное письмо:

```
$to = "david@gowansnet.com";
$subject = "Чудесный PHP";
$body = "PHP - один из самых лучших скриптовых языков";
$headers = "From: webmaster@gowansnet.com\n";
mail($to,$subject,$body,$headers);
echo "Письмо отправлено по адресу $to";
```

Этот код делает две вещи. Во-первых, он отпраляет письмо по адресу david@gowansnet.com с темой "Чудесный PHP" и текстом:

```
PHP - один из самых лучших скриптовых языков
```

Письмо придет с адреса webmaster@gowansnet.com. Кроме того, он выведет в браузере текст:

```
Письмо отправлено по адресу david@gowansnet.com
```

Форматирование E-mail

Возможно, вы заметили, что срока, в которой указывается содержание поля "From" заканчивается символами \n. Это очень важные символы, когда мы отправляем e-mail. Это символы начала новой строки, они указывают PHP сделать в e-mail новую строку. Очень важно, чтобы перевод строки происходил после каждого e-mail-заголовка, только тогда письмо будет соответствовать международным стандартам и будет доставлено на любой адрес.

Символы \n также можно применять для перевода строки в разделе body (в тексте письма), но их нельзя вставлять в разделы subject или To.

Отправка почты без переменных

Выше приведенный e-mail можно отправить, используя разные имена переменных (имеет значение не название переменных, а их положение в команде mail()). Можно отправить письмо кодом, состоящим всего из одной строки, без использования переменных:

```
mail("david@gowansnet.com",
"Чудесный PHP",
"PHP - один из самых лучших скриптовых языков",
"From: webmaster@gowansnet.com\n");
```

Но такой код немного труднее читать.

Контроль ошибок

Каждый, кто имеет дело со скриптами, знает, что в коде очень легко допустить ошибку, часто делают ошибки и при введении e-mail-адреса (особенно, когда он вносится в форму и обрабатывается скриптом). По этой причине можно добавить небольшой кусочек кода, который будет проверять успешную отправку e-mail:

```
if(mail($to,$subject,$body,$headers)) {
echo "Письмо было отправлено по адресу $to с темой:
$subject";
} else {
echo "С отправкой письма возникли проблемы.
Проверьте код и убедитесь, что e-mail-адрес
$to правильный";
}
```

Здесь все понятно. Если письмо отправлено успешно, в браузере будет выведено сообщение об успешной отправке. Если нет, будет выведено сообщение об ошибке и о возможных ее причинах.

Работа с формами

Конструирование формы

Конструирование формы для работы с PHP-скриптом ничем не отличается от создания обычной HTML-формы. Поскольку это школа по языку PHP, мы не станем сильно углубляться в правила создания форм, но мы познакомимся с тремя основными частями кода, которые нужно знать:

```
<input type="text" name="thebox" value="Ваше имя">
```

Этот код выведет бокс ввода текста, в котором по умолчанию будет вписано ваше имя. Значение атрибута value здесь - произвольно. Информация же, заданная в атрибуте name является уникальным именем-идентификатором этого текстового бокса.

```
<textarea name="message">
Здесь пишите свое сообщение.
</textarea>
```

Этот код отобразит большой бокс текстового ввода с прокруткой, в нем по умолчанию будет вписан текст "Здесь пишите свое сообщение." И снова, атрибут name задает уникальное имя-идентификатор этого бокса.

```
<input type="submit" value="Submit">
```

Этот код отобразит кнопку типа submit. Можно изменить то, что на ней написано, изменив значение атрибута value.

Все элементы формы должны быть заключены в тэги <form>. Это делается так:

```
<form action="process.php" method="post">
Элементы формы, форматирование и т.д.
</form>
```

Атрибут формы action определяет какому скрипту отправляются данные из формы (в данном случае process.php). Тут также может быть вписан полный адрес скрипта-обработчика URL (например, http://www.mysite.com/scripts/processors/process.php). Атрибут method определяет, каким образом отправляются данные. Если его значение POST, то данные предоставляются скрипту в поток данных по запросу. Если его значение GET, данные отправляются в виде адреса URL, где они идут после вопросительного знака, например, http://www.mysite.com/process.php?name=david

На самом деле нет никакой разницы, какой способ отправки данных применять, но если вы передаете пароли или другую важную информацию, лучше использовать метод POST, тогда передаваемые данные не будут видны в адресной строке браузера.

Получение информации из формы

Следующий шаг - получение скриптом информации, введенной в вашу форму, чтобы он мог с нею что-нибудь делать. В PHP, в отличие от многих других языков, это сделать очень просто. Говоря в общем, полученные данные можно использовать в скрипте, указывая имена элементов формы на странице. Например, выше был пример с текстовым боксом, имеющим имя-идентификатор "message". Если вы хотите использовать данные из этого текстового поля в своем скрипте, вам нужно использовать переменную $message, которая будет содержать нужные данные. Только нужно быть внимательным и не переписать значение этой переменной в своем скрипте на какое-то другое.

Еще об одном моменте следует здесь сказать. Вы можете отправлять скрипту информацию в виде стандартного URL. Тот же принцип, что и отправка данных с помощью метода GET. Это позволяет создавать различные ссылки на один и тот же скрипт, которые делают разные вещи. Например, можно создать скрипт, который будет показывать различные страницы, в зависимости от того, по какой ссылке пошли:

```
yourpage.php?user=david
```

отобразит страницу Дэвида,

```
yourpage.php?user=tom
```

тот ж самый скрипт отобразит страницу Тома.

Как говорилось выше, в скрипте вы можете обращаться к переменной $user.

Также по той же системе можно отправлять скрипту несколько кусков информации, разделяя каждый кусок амперсандом &:

```
yourpage.php?user=david&referrer=gowansnet&area=6
```

В результате в скрипте-обработчике можно будет обращаться к независимым переменным $user, $referrer и $area.

Создание формы для почтового скрипта

В заключение этого раздела, мы познакомимся с тем, как можно использовать все это для создания системы, которая отправит вам по e-mail коментарии пользователя.

Во-первых, создадим на своей HTML-странице вот такую форму:

```
<form action="mail.php" method="post">
Ваше имя: <input type="text" name="name"><br>
E-mail: <input type="text" name = "email"><br><br>
Коментарии<br>
<textarea name="comments"></textarea><br><br>
<input type="submit" value="Отправить">
</form>
```

В результате на странице отобразится простая форма, в которой пользователь сможет ввести свое имя, свой адрес e-mail и сам коментарий. Конечно, вы можете добавить в свою форму и другие элементы, но не забудьте тогда и дополнить сам скрипт-обработчик. Теперь напишем PHP-скрипт:

```
<?
$to="webmaster@gowansnet.com";
$message="$name только что оставил свой коментарий.
Он написал:\n$comments\n\nЕго адрес e-mail: $email";
if(mail($to,'Коментарий о вашем сайте',$message,
'From: $email\n')) {
echo "Спасибо за ваш коментарий."
} else {
echo "Есть проблемы с отправкой сообщения.
Првоверьте правильность заполнения формы.";
}
?>
```

Не забудьте заменить адрес webmaster@gowansnet.com на свой собственный e-mail. Этот скрипт нужно сохранить как mail.php и выложить на свой сервер. И все, можно заполнять форму коментария.

Заключение

Некоторые дополнительные вопросы, не заслуживающих особого раздела

Коментарии

Как и влюбом другом языке программирования, в свой скрипт очень важно вставлять комментарии. Если со скриптом работаете не только вы, нужно оставлять своим партнерам возможность понять что делает та или иная часть кода, а если вы распространяете свой скрипт, нужно, чтобы люди знали, как его редактировать. Даже если над скриптом работаете только вы, полезно оставлять коментарии, чтобы позже вы смогли в нем разобраться.

В PHP есть два способа размещения коментариев. Первый способ применяется для коментариев, которые умещаются в одну строку. Второй используется для длинных многострочных коментариев. Однострочный коментарий пишется так:

```
// Здесь размещается коментарий
```

Все, что располагается после символов // будет пропускаться во время выполнения скрипта. Так можно размещать коментарии на одной строке с другим кодом, например:

```
print "Hello $name"; // Приветствие пользователю
```

А вот как размещаются многострочные комментарии:

```
/* Далее идет код, который принимает данные у
пользователя и проверяет их валидность
перед размещением в базе данных*/
```

Все, что расположено между символами /* и */ будет проигнорировано. Только не забывайте закрывать этот тип коментария, иначе скрипт может не работать.

Команды Print, Echo и HTML

Как вы могли заметить, на этой школе мы использовали 4 различных способа выведения информации в окне браузера:

```
echo("Text here");
echo "Text here";
print("Text here");
print "Text here";
```

Все эти способы делают одно и то же и в своих скриптах можно пользоваться любым. Даже нет никакой причины пользоваться в скрипте одним и тем же способом. Единственная проблема в том, что перед кавычками " в HTML-коде должны расставляться обратные слэши: \". Если код длинный, то расставлять обратные слэши довольно муторно. И тут мы подходим к очень полезной возможности PHP. Если, например, вы создаете заголовок страницы динамически с помощью PHP, затем идет статическое содержание страницы и все заканчивается динамически создаваемым футером, можно сделать так:

```
<?
PHP-код динамического заголовка
?>
Статичный HTML-код
<?
PHP-код динамического футера
?>
```

Более того, PHP-код продолжает выполняться с того места, на котором он оборвался, так что можно его разрывать даже так:

```
<?
Выражение IF {
?>
HTML-код, если условие IF выполняется
<?
} else {
?>
HTML-код, если условие IF не выполняется
<?
}
?>
```

Помните, что всегда нужно закрывать выражения IF и циклы - это легко забыть.

Запись команд в одну строку

Помещение HTML-кода в PHP-скрипт очень полезно, но как быть, если нужно в HTML-код поместить значение переменной? Здесь нельзя, как при использовании команд echo или print, просто вставить имя переменной, поскольку HTML-код не является обрабатываемой частью

PHP-скрипта. Вместо этого нужно в сам HTML-код вставить небольшой кусочек PHP.

Например, если вы хотите вывести с помощью скрипта чье-либо имя, отформатированное HTML-тэгами, нужно сделать так:

```
<font face="Arial" size="7" color="red">
<b><? echo($variablename); ?></b></font>
```

В этом HTML-коде встроен небольшой кусочек PHP:

```
<? echo($variablename); ?>
```

Эта запись совершенно аналогична следующей:

```
<?
echo($variablename);
?>
```

но записана в одну строку.

Заключение

На этой школе вы получили основные знания о языке PHP, которые позволят вам сделать большую часть того, что вы задумали. Более глубокие знания ищите на PHP.net[1], официальном сайте языка PHP. Может быть, вы обратили внимание, что на этой школе мы не затрагивали тему применения PHP при работе с базами данных. Поскольку именно здесь PHP применяется наиболее широко и тут есть много возможностей, эта тема вынесена в отдельную школу PHP/MySQL. Добро пожаловать на нашу школу PHP/MySQL![19] (open link)

Введение

Предисловие

Для многих людей главной причиной, побуждающей их к изучению скриптовых языков, подобных PHP, являются возможности, которые они открывают при работе с базами данных. На этой школе мы познакомимся с тем, как можно использовать PHP и базу данных MySQL для хранения информации в Интернете и размещения ее на ваших веб-страницах. Прежде, чем вы начнете читать этот учебник, вам нужно получить хотя бы общее представление о языке PHP. Если вы его еще не знаете совсем, лучше сначала отправиться на школу PHP[20] (open link).

Зачем нужна база данных?

Удивительно, насколько полезной может быть база данных при создании веб-сайта. Существует огромное количество применений баз данных в этой области: от отображения простых списков до полной генерации целого сайта из базы данных. Вот лишь некоторые возможности, которые открывает совместное использование PHP и MySQL:

1: http://www.php.net/
19: http://xml.nsu.ru/php/php2_intro.xml
20: http://xml.nsu.ru/php/php_intro.xml

- **Управление баннерами**. Там, где должен быть баннер, вызывается PHP-скрипт. Он открывает базу данных, выбирает из нее случайный баннер и показывает его визитеру. Кроме того, он может подсчитывать общее число показов каждого баннера и, возможно, число кликов по нему. Для того, чтобы добавить, изменить или отредактировать баннеры, мне нужно только изменить базу данных, а скрипт позаботится обо все остальном.
- **Форумы**. В Интернете сотни форумов работают на основе PHP и MySQL. Это оказывается гораздо эффективнее других систем, в которых для каждого сообщения создается отдельная страница, и предоставляет большое количество дополнительных возможностей. Все страницы форума можно обновлять, производя изменения в одном скрипте.
- **Базы данных**. Один из самых очевидных примеров - это сайты, которое все свое содержание получают из базы данных. Например, сайт Script Avenue работает на основе нескольких скриптов, которые получают всю информацию, которая идет на страницы, из большой базы данных. К различным категориям скриптов, распространяемых на этом сайте, можно получить доступ с помощью одного-единственного скрипта, просто изменяя URL - и вы обратитесь к различным разделам этой базы данных.
- **Веб-сайты**. Если у вас - большой сайт и вы хотите поменять его дизайн, можно потратить на это очень много времени: сначала нужно переделать каждую страницу, затем выложить все это на сервер... С помощью PHP и MySQL весь ваш сайт может создаваться лишь одним или двумя PHP-скриптами. Они просто обращаются к базе данных MySQL и получают из нее информацию для веб-страниц. Для того, чтобы поменять дизайн всего сайта, нужно поменять только шаблон, по которому строятся страницы.

Что для этого нужно?

Для того, чтобы применять PHP-скрипты, работающие совместно с базой данных MySQL, нужно всего три вещи. Во-первых, вам, конечно, нужен веб-сервер. Он может размещаться либо на вашем компьютере, либо на веб-хосте. Вместе с PHP и MySQL может работать любое серверное программное обеспечение, но лучше всего использовать Apache, который к тому же бесплатный.

Кроме того, на вашем сервере должен быть установлен PHP. Если он еще не установлен, вам нужно его установить (или попросить, чтобы его установили на вашем веб-хосте). Его можно скачать с PHP.net, он тоже распространяется бесплатно. Если вы не знаете, установлен ли на вашем сервере PHP, ниже мы познакомимся с тем, как это проверить.

И наконец, вам понадобится база данных MySQL. Это программное обеспечение для создания и поддержки баз данных. Вы конечно можете использовать и большинство других типов баз данных (SQL, Oracle и т.д.), но поскольку это школа PHP/MySQL, мы будем говорить о базах данных MySQL (хотя команды, которые мы будем изучать, можно применять и при работе с базами данных SQL). Как и все программы, которыми мы будем пользоваться, MySQL - бесплатна и ее можно скачать с официальной страницы MySQL. Если вы не знаете, установлена ли у вас MySQL, ниже мы познакомимся с тем, как это проверить.

Проверка наличия PHP и MySQL

Вот простой способ проверить, установлены ли на сервере PHP и MySQL. Откройте текстовый редактор и напишите следующее:

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

Сохраните этот код как phpinfo.php

Теперь выложите этот файл на ваш сервер и откройте его в браузере. Если PHP установлен, вы увидите большую страницу, на которой будет много разных таблиц, содержащих информацию об установленном на сервере PHP. Теперь прокрутите страницу вниз. Если вы найдете раздел, посвященный MySQL, значит MySQL на этом сервере установлена.

Управление базами данных

Хотя с помощью PHP-скриптов можно производить все администрирование вашей базы данных, очень полезно на свой сервер установить PHPMyAdmin. Это прекрасный набор скриптов, которые предоставят вам административный интерфейс для управления вашей базой данных MySQL. С помощью него вы сможете добавлять, удалять, редактировать и делать резервные копии своих баз данных, и особенно он полезен для устранения неполадок в базах данных.

Наша школа

На этой школе мы познакомимся с основами совместного использования PHP и MySQL. Для этого мы через все разделы будем работать с одним примером. Вы узнаете как создать основанную на веб-технологии программу для работы с бизнес-контактами. Она позволит вам сохранять имена, адреса, e-mail и телефонные номера. Вы сможете обновлять эти записи, производить поиск по базе данных. Кроме того, будет возможность организации почтовой рассылки персонам, занесенным в вашу базу данных (только не используйте, пожалуйста, эту систему для рассылки спама на случайные адреса).

После создания этой системы у вас будет достаточно знаний для дальнейших своих собственных разработок, например, сайта, полностью генерируемого из базы данных.

Создание базы данных

Создание базы данных в MySQL и ее подготовка к приему данных

Строение базы данных

Базы данных MySQL имеют стандартное устройство. Они представляют собой базу данных, в которой содержатся таблицы. Каждая из этих таблиц - самостоятельна и может, например, состоять из разных полей и т.д., хотя все они являются частью одной базы данных. Каждая таблица содержит записи (records), которые состоят из полей (fields).

Базы данных и логины

Процесс создания базы данных MySQL может варьирорвать от хоста к хосту, но в конечном итоге у вас будет имя базы данных, имя пользователя и пароль. Для соединения с базой данных будет нужна эта информация.

Если вы используете PHPMyAdmin (или схожую программу), то вы просто ее запускаете, вводите имя пользователя и пароль и присоединяетесь к своей базе данных. Если нет - вы должны производить все администрирование своей базы данных с помощью PHP-скриптов.

Создание таблицы

Прежде, чем вы сможете что-то сделать со своей базой данных, вам нужно создать таблицу. Таблица - это раздел базы данных для хранения структурированной информации. В таблице задаются различные поля. Благодаря такому устройству, почти все требования, которые предъявляет веб-сайт к базе данных, можно выполнить с помощью одной-единственной базы данных.

С помощью PHPMyAdmin создать таблицу просто: впишите имя, выберете число полей и нажмите на кнопку. После этого вы окажитесь на странице, где вы будете создавать поля для своей базы данных. Если для создания базы данных вы используете PHP-скрипт, весь процесс создания и организации можно выполнить одной командой.

Поля

В MySQL имеется большое количество различных типов полей и возможных атрибутов, здесь мы познакомимся только с некоторыми:

Тип поля	Описание
TINYINT	Маленькое целое число
SMALLINT	Маленькое целое число
MEDIUMINT	Целое число
INT	Целое число
VARCHAR	Текст (длиной не больше 256 символов)
TEXT	Текст

Это только некоторые из возможных типов полей. Поищите в Интернете полный список доступных типов полей.

Создание таблицы с помощью PHP

Создать таблицу в PHP только лишь немного труднее, чем с PHPMyAdmin. Используется следующий код:

```
CREATE TABLE имя_таблицы {

Поля


}
```

Поля задаются следующим образом:

```
имя_поля тип_поля(длина) дополнительная_информация,
```

Последнее задание поля не должно иметь в конце запятой.

С развернутым примером создания таблицы мы познакомимся ниже.

База данных бизнес-контактов

База данных бизнес-контактов будет хранить всю контактную информацию о важных для вашего бизнеса людях и эту информацию можно будет просматривать и редактировать через Интернет. В этой базе данных будут иметься следующие поля:

Имя	Тип	Длина	Описание
id	INT	6	Уникальный идентификатор каждой записи
first	VARCHAR	15	Фамилия
last	VARCHAR	15	Имя
phone	VARCHAR	20	Номер телефона
mobile	VARCHAR	20	Номер мобильного телефона
fax	VARCHAR	20	Номер факса
email	VARCHAR	30	Адрес e-mail
web	VARCHAR	30	Адрес веб-страницы

Вам, может быть, непонятно, почему мы для полей с номерами телефона и факса взяли тип VARCHAR, ведь эта информация - цифровая. Конечно, здесь можно использовать тип поля INT, но лучше применять все же тип VARCHAR, потому что такой тип позволяет в ряду цифр в номерах оставлять пробелы, дефисы и вставки текста (например, 1800-COMPANY). Если мы не производим телефонные звонки через Интернет, это не создаст никаких проблем.

Еще на одну вещь нужно обратить внимание в этой базе данных. Поле id должно быть определено как PRIMARY, INDEX, UNIQUE и должно быть установлено на auto_increment (В программе PHPMyAdmin - в меню Extra). Это потому, что это поле - первичное и индексное, поэтому должно быть уникальным. Установка auto_increment (автоматическое приращение) означает, что всякий раз, когда вы добавляете запись, если вы не задаете какой-то конкреный id, этот идентификатор получит следующее за последним значение.

Если вы используете PHPMyAdmin или другую программу администрирования, вы можете все это сделать в таблице, имеющей название contacts.

Создание таблицы в PHP

Для создания таблицы в PHP следует использовать следующий код. Кое-что в этом коде еще не было объяснено, но в следующем разделе мы это сделаем.

```
<?
$user="username";
$password="password";
$database="database";
mysql_connect(localhost,$user,$password);
@mysql_select_db($database) or
die( "Unable to select database");
$query="CREATE TABLE contacts
(id int(6) NOT NULL auto_increment,
```

```
first varchar(15) NOT NULL,
last varchar(15) NOT NULL,
phone varchar(20) NOT NULL,
mobile varchar(20) NOT NULL,
fax varchar(20) NOT NULL,
email varchar(30) NOT NULL,
web varchar(30) NOT NULL,
PRIMARY KEY (id),UNIQUE id (id),KEY id_2 (id))";
mysql_query($query);
mysql_close();
?>
```

Впишите имя базы данных, имя пользователя MySQL и пароль MySQL на соответствующие места в первых трех строчках этого кода.

Вставка информации

В этом разделе мы узнаем, как вставлять информацию в созданную базу данных

Присоединение к базе данных

Прежде, чем начинать работу, нужно присоединиться к базе данных MySQL. Это - чрезвычайно важный этап, поскольку если вы не присоединитесь, команды к базе данных не пройдут.

Хорошая привычка при работе с базами данных - задание имени пользователя, пароля и имени базы данных в виде переменных, так что если со временем вам понадобится что-то изменить, нужно будет переписать только одну строчку кода:

```
$username="username";
$password="password";
$database="your_database";
```

Здесь вы можете подумать о том, что хранить свой пароль в файле небезопасно. Тут вам не нужно волноваться, потому что перед отправкой пользователю PHP-код обрабатывается сервером, так что пользователь не может увидеть исходный код скрипта.

Далее нужна команда на присоединение к базе данных:

```
mysql_connect(localhost,$username,$password);
```

В этой строке PHP-скрипту дается указание присоединиться к серверу, несущую базу данных MySQL, расположенному на 'localhost', используя имя пользователя, хранящееся в переменной $username и пароль, хранящийся в переменной $password.

Прежде, чем мы узнаем, как работать с базой данных, познакомимся еще с одной командой:

```
mysql_close();
```

Это очень важная команда, она закрывает соединение с сервером базы данных. Ваш скрипт будет работать и дальше, если вы не включите в него эту команду, но много незакрытых соединений с базой данных MySQL могут вызвать проблемы на веб-хосте Хорошая привычка - вставлять эту команду после всех остальных команд работы с базой данных, чтобы хост работал как надо.

Выбор базы данных

После присоединения к серверу базы данных, вам нужно выбрать базу данных, с которой вы будете работать. Это должна быть база, которой соответствует имя пользователя. Для этого используется следующая команда:

```
@mysql_select_db($database) or
die( "Unable to select database");
```

Она сообщает PHP-скрипту, что нужно выбрать базу данных, имя которой хранится в переменной $database (ее вы задали выше). Если соединение установить не удается, скрипт останавливается и выводит текст о том, что выбрать нужную базу данных не удается:

```
Unable to select database
```

Unable to select database

Эту часть кода (or die) можно применять для получения небольшого контроля за ошибками, но вообще-то эта часть не существенна.

Выполнение команд

Теперь, когда вы присоединились к базе данных и выбрали нужную, можно начать выполнять на сервере команды.

Существует два способа выполнения команд. Первый способ - простое введение команд в PHP. Этот способ применяется, когда у операций не будет выходного результата.

Второй способ - задание команды как переменной. В этом способе создается переменная с результатами операции.

В этом разделе школы мы будем применять первый способ, поскольку мы пока не ожидаем никакого выходного результата от базы данных. Команда выглядит так:

```
mysql_query($query);
```

У этой формы команды есть то преимущество, что вы можете снова и снова применять одну и ту же команду mysql_query(); и не изучать новые. Нужно только менять значение переменной.

Вставка данных

В этом разделе мы вернемся к базе данных бизнес-контактов. Сейчас мы добавим в нее первую информацию:

Field	Information
Фамилия:	Smith
Имя:	John
Телефон:	01234 567890
Мобильный телефон:	00112 334455
Факс:	01234 567891
E-mail:	johnsmith@gowansnet.com

Веб-страница: http://www.gowansnet.com

Всю эту информацию мы введем одной командой:

```
$query = "INSERT INTO contacts VALUES
('','Smith','John','01234 567890','00112 334455',
'01234 567891','johnsmith@gowansnet.com',
'http://www.gowansnet.com')";
```

Расшифруме, что все это значит.

Во-первых, мы задаем переменную $query, которая будет содержать нашу команду (смотри предыдущий параграф). Смысл следующего куска понять не сложно:

```
INSERT INTO contacts VALUES
```

В нем PHP-скрипту дается указание вставить в таблицу, имеющую имя contacts значения, заключенные в скобки, они идут далее.

Кусок кода в скобках содержит всю добавляемую информацию. В нем все поля идут по порядку, значение каждого поля заключено в одинарные кавычки. Например:

```
John
```

будет вставлено во третье поле, которое в таблице соответствует имени человека.

Моджет быть, вы заметили, что мы не вставляем информацию в первое поле таблицы (id). Это потому, что это поле выступает в роли индексного. В базе данных не может быть двух записей, имеющий одинаковый ID. Поэтому когда мы создавали базу данных, мы установили это поле на 'auto_increment', автоматическое приращение, то есть значение этого поля будет приращением на единицу ID предыдущей записи. При этом первая запись будет иметь ID равный 1.

Отображение данных

Создание страницы ввода данных в базу и отображение содержимого базы данных

HTML-форма ввода

Введение данных через HTML-страницы почти идентично вставке данных с помощью PHP-скрипта. Однако, преимущество состоит в том, что вам не нужно изменять скрипт под каждую вводимую в базу информацию и вы можете позволить всем пользователям вводить свои собственные данные.

Следующий код отобразит HTML-страницу, на которой расположены поля текстового ввода для введения нужной информации:

```
<form action="insert.php" method="post">
Фамилия: <input type="text" name="first"><br>
Имя: <input type="text" name="last"><br>
Телефон: <input type="text" name="phone"><br>
Мобильный телефон: <input type="text" name="mobile"><br>
Факс: <input type="text" name="fax"><br>
E-mail: <input type="text" name="email"><br>
Веб-страница: <input type="text" name="web"><br>
<input type="Submit">
```

```
</form>
```

Конечно, эта страница может быть иначе отформатирована и в ней можно изменить другие детали. Это просто базовая форма, от которой можно работать. Теперь нужно отредактировать скрипт из предыдущего раздла. Вместо использования для вставки данных в базу самой информации, мы будем применять переменные:

```
<?
$username="username";
$password="password";
$database="your_database";

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or
die( "Unable to select database");

$query = "INSERT INTO contacts VALUES
('','$first','$last','$phone','$mobile','$fax',
'$email','$web')";
mysql_query($query);

mysql_close();
?>
```

Теперь этот скрипт можно сохранить как input.php и его можно вызывать из HTML-формы. Это позволит вводить данные не только локально, но вводить их в форму, они будут сохраняться как переменные, которые передаются PHP-скрипту.

Еще можно вставить в скрипт выведение сообщения о том, что ввод данных был успешен.

Вывод данных

Теперь у вас уже есть в базе хотя бы одна запись. Как, используя PHP, выводить данные? Прежде, чем вы это узнаете, вам нужно познакомиться с циклами в PHP (смотри школу PHP), поскольку здесь они применяются.

Первая команда, которая нам потребуется - это команда MySQL query, составленная так:

```
SELECT * FROM contacts
```

Это основная команда MySQL, которая дает скрипту указание выбрать все записи в таблице contacts. Поскольку эта команда подразумевает выводимый результат, его нужно сохранить в переменной:

```
$query="SELECT * FROM contacts";
$result=mysql_query($query);
```

В данном случае все содержимое базы данных будет передано специальному массиву, имеющему имя $result. Прежде, чем вывести данные, нужно преобразовать его в отдельные переменные. Этот процесс состоит из двух стадий.

Подсчет рядов

Прежде, чем вы выведете данные, нужно знать, сколько рядов имеется в вашей базе данных. Конечно, число рядов можно просто ввести в код, но хорошо бы сделать, чтобы не приходилось каждый раз менять скрипт, когда число рядов изменяется. Вместо этого можно исполь-

зовать следующую команду:

```
$num=mysql_numrows($result);
```

Здесь создается переменная $num и ее значение устанавливается равным числу рядов, хранимых в массиве $result (результате, полученном из базы данных). Теперь для получения данных и их вывода на экран можно применить цикл.

Организация цикла

Нужно организовать цикл, который возьмет каждый ряд результата и выведет данные, в нем находящиеся. Используя переменную $num, можно легко организовать цикл через все ряды. В ниже приведенном коде $i содержит число раз, которые отработал цикл, эта переменная служит для того, чтобы цикл остановился с последним обработанным рядом:

```
$i=0;
while ($i < $num) {

код

++$i;
}
```

Это обычный цикл в PHP и он выполнит заданный код нужное число раз. При этом после каждого прохода цикла $i будет увеличиваться на 1. Поэтому переменную $i можно использовать для определения какой ряд следует читать скрипту. Поскольку первая строка выводимых данных в MySQL имеет номер 0, все будет правильно работать.

Распределение данных по переменным

Последняя часть скрипта для вывода данных - сохранение каждого куска данных в отдельной переменной. Для этого применяется следующий код:

```
$variable=mysql_result($result,$i,"fieldname");
```

Так что для выделения каждого отдельного кусочка информации в нашей базе данных, нужно сделать так:

```
$first=mysql_result($result,$i,"first");
$last=mysql_result($result,$i,"last");
$phone=mysql_result($result,$i,"phone");
$mobile=mysql_result($result,$i,"mobile");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");
```

Нам не нужно получать значение поля ID (хотя его можно получить), потому что мы его не используем в нашей странице вывода.

Объединение скрипта

Теперь мы можем записать полный скрипт для выведения данных. В этом скрипте выводимые данные никак не форматируются:

```
<?
```

```
$username="username";
$password="password";
$database="your_database";

mysql_connect(localhost,$username,$password);
@mysql_select_db($database) or
die( "Unable to select database");
$query="SELECT * FROM contacts";
$result=mysql_query($query);

$num=mysql_numrows($result);

mysql_close();

echo "<b><center>Database Output
</center></b><br><br>";

$i=0;
while ($i < $num) {

$first=mysql_result($result,$i,"first");
$last=mysql_result($result,$i,"last");
$phone=mysql_result($result,$i,"phone");
$mobile=mysql_result($result,$i,"mobile");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");

echo "<b>$first $last</b><br>
Телефон: $phone<br>
Мобильный телефон: $mobile<br>
Факс: $fax<br>
E-mail: $email<br>
Веб-страница: $web<br>
<hr><br>";

++$i;
}

?>
```

Другие способы вывода

Другие способы выведения и отображения информации из базы данных

Форматирование выходных данных

В предыдущем разделе мы научились выводить из базы данных контактную информацию всех людей, имеющихся в ней. Мы вывели эту информацию в самом общем виде, не особенно подходящем для действующего веб-сайта. Хорошо бы научиться форматировать выходные данные и выводить их в виде таблицы.

Произвести такого рода форматирование не очень сложно. Для этого нужно лишь применить

PHP для выведения соответствующего HTML-кода и при этом в определенных его местах вызывать соответствующие переменные. Простейший способ - закрыть PHP-тэг и обычным способом вводить HTML. При достижении позиции, где должна быть отображена переменная, вставьте следующий код:

```
<? echo "$variablename"; ?>
```

Также можно применять PHP-цикл для повторения нужного кусочка кода и включения его в качестве составной части HTML-описания таблицы. Например, используя приводившийся выше код, который циклически выводил данные, можно отформатировать их в одну большую таблицу:

```
<table border="0" cellspacing="2" cellpadding="2">
<tr>
<th>
<font face="Arial, Helvetica, sans-serif">Имя</font>
</th>
<th>
<font face="Arial, Helvetica, sans-serif">Телефон</font>
</th>
<th>
<font face="Arial, Helvetica, sans-serif">
Мобильный телефон</font>
</th>
<th>
<font face="Arial, Helvetica, sans-serif">Факс</font>
</th>
<th>
<font face="Arial, Helvetica, sans-serif">E-mail</font>
</th>
<th>
<font face="Arial, Helvetica, sans-serif">
Веб-сайт</font>
</th>
</tr>

$i=0;
while
($i < $num) {

$first=mysql_result($result,$i,"first");
$last=mysql_result($result,$i,"last");
$phone=mysql_result($result,$i,"phone");
$mobile=mysql_result($result,$i,"mobile");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");
?>

<tr>
<td><font face="Arial, Helvetica, sans-serif">
<? echo "$first $last"; ?></font></td>
<td><font face="Arial, Helvetica, sans-serif">
<? echo "$phone"; ?></font></td>
<td><font face="Arial, Helvetica, sans-serif">
<? echo "$mobile"; ?></font></td>
```

```
<td><font face="Arial, Helvetica, sans-serif">
<? echo "$fax"; ?></font></td>
<td><font face="Arial, Helvetica, sans-serif">
<a href="mailto:<? echo "$email"; ?>">E-mail</a>
</font></td>
<td><font face="Arial, Helvetica, sans-serif">
<a href="<? echo "$web"; ?>">Веб-сайт</a>
</font></td>
</tr>

<?
++$i;
}

echo "</table>";
```

Этот код сначала выведет заголовки таблицы, затем будет добавлять в таблицу по одному ряду для каждой записи, форматируя выходные данные.

Если вы уже немного знакомы с PHP и HTML, разобраться в этом коде вам будет нетрудно, мы только обратим внимание на какую-нибудь строку в таблице, например:

```
<a href="mailto:<? echo "$email"; ?>">E-mail</a>
```

Здесь видна легкость применения PHP для внедрения в страницы данных из MySQL, таким образом можно делать страницы полностью динамическими.

Выбор определенных данных

PHP можно применять не только для показа содержимого всей базы данных, но и для выбора отдельных записей или записей, отвечающих какому-нибудь критерию. Для этого применяется видоизмененный запрос SELECT. Для показа всей таблицы мы применяли запрос:

```
SELECT * FROM contacts
```

Если же мы хотим выбрать те записи, в которых фамилия человека - Smith, следует применять следующий запрос:

```
SELECT * FROM contacts WHERE first='Smith'
```

Как и другие запросы в MySQL, этот выглядит почти как обычный английский текст. Таким же образом можно сделать запрос и на основе другого поля в таблице. Можно также сделать запрос и по нескольким полям, добавив в запрос дополнительную секцию:

```
поле='значение'
```

Мы не станем здесь вдаваться слишком глубоко в эту тему, но для указания в запросе критериев поиска можно применять и переменные. Например, если вы создали форму для поиска в базе данных, вы можете сохранить имена людей в переменной $searchlast. Затем нужно выполнить следующий код:

```
$query="SELECT * FROM contacts
WHERE last='$searchlast'";
$result=mysql_query($query);
```

Обратите внимание, что в конце первой строки идет одинарная кавычка ', за которой идет двойная кавычка ", а затем - точка с запятой.

Отдельные записи и обработка ошибок

Дополнительные сведения об отображении информации: выбор определенных данных, предотвращение ошибок при выводе

Обработка ошибок

Когда вы выводите все данные из базы данных, вряд ли может так случиться, что в ней вообще не окажется данных, но если вы ее обновляете и удаляете записи, это вполне может случиться. К счастью, при использовании PHP и MySQL, существует простой способ избегать ошибок вывода в этом случае:

```
$num=mysql_numrows($result);
```

Здесь переменная $result содержит результат запроса в базу данных (например, выбраны все записи в базе). Как мы знаем, эта строка присвоит переменной $num число рядов в результате запроса (эта переменная нами использовалась в при организации цикла через результат запроса). Теперь вы можете организоватиь простую обработку ошибки, используя выражение IF:

```
if ($num==0) {
echo "В базе данных не содержится информации";
} else {
Цикл вывода результата запроса
}
```

Вы можете развить этот пример и сделать его более дружелюбным к пользователю (например, если в базе данных нет ничего, дать ссылку на страницу введения информации в базу данных).

Упорядочивание данных

Можно выводить данные не только по содержимому отдельных полей, можно также упорядочивать выводимые данные по соедержимому полей (например, сортируя выводимые записи по алфавитному порядку). По умолчанию, записи выводятся в порядке возрастания содержимого поля id: от 1 и далее, в порядке возрастания. Но можно сортировать и по любому другому полю.

Например, полезно рассортировать все выводимые записи по фамилиям людей в результате запроса. Для тех, кто не знаком с работой обычных баз данных, скажем, что это достигается сортировкой в порядке возрастания поля "фамилия": A до Z. (Цифры в порядке возрастания выводятся от 1 до 10 и т.д., в порядке убывания - от Z до A и от 10 до 1). Для этого можно использовать следующий запрос:

```
SELECT * FROM contacts ORDER BY first ASC
```

Для выведения записей в порядке убывания фамилии, нужно использовать параметр DESC.

Другие применения mysql_numrows и сортировки

Полученное нами значение $num очень важно, потому что, кроме обработки ошибок и организации циклов, у нее могут быть и другие применения. Например, мы хотим вывести только

последние пять записей, введенных в базу данных. Во первых, записи нужно рассортировать по значению поля id (поскольку самые последние введенные записи имеют самый большой ID). Лучше всего их рассортировать в порядке убывания ID.

Теперь записи расположены по порядку от самых последних - к самым старым, но нам нужно выделить пять самых последних. Для этого нужно вывести циклом не все записи, (их количество содержится в переменной $num), а только 5 из них, для этого цикл вывода данных должен сработать только 5 раз.

Конечно, перед этим нужно проверить, превышает ли $num число 5, потому что если цикл сработает пять раз, а в базе всего три записи, произойдет ошибка. Это сделать не трудно, например, следующим кодом:

```
if ($num>5) {
$to=$num;
}else{
$to=5;
}

$i=0;
while ($i < $to) {
Остальной код
```

Этот код проверит, что в базе данных более, чем 5 записей. Если этот так, сработает пять раз цикл вывода, если же их меньше, цикл сработает правильное число раз и выведет всю базу данных.

Поле ID

Если вы вспомните тот раздел этой школы, в коотром описывалось создание базы данных, вы припомните, что мы включили в нее числовое поле с названием id. Это поле было установлено нами на авто-приращение и было указано, что это - первичное поле. Мы уже обсудили, почему значение этого поля оказывается уникальным для каждой записи в базе, а сейчас бы сделаем следующий шаг и узнаем, как это поле можно использовать для выбора отдельных записей из базы данных.

Выбор отдельной записи

В предыдущем разделе этой школы мы познакомились с тем, как выбирать из базы данных записи, основываясь на содержимом определенных полей:

```
SELECT * FROM contacts WHERE поле='значение'
```

Теперь, используя поле с уникальным ID каждой записи, мы можем выбрать из базы данных любую запись:

```
SELECT * FROM contacts WHERE id='$id'
```

Здесь $id - переменная, содержащая номер записи. Казалось бы, для чего может понадобиться выбирать записи по их номеру? Но это может пригодиться в нескольких случаях. Например, если вы создаете динамический сайт, геренирующийся с помощью единственного PHP-скрипта из базы данных, вам понадобится создать скрипт, который встраивает в страницу содержимое базы данных. Затем, с помощью поля id, вы можете выбрать в базе содержимое каждой отдельной страницы сайта и вывести ее. Можно даже использовать URL старницы для нужной вам записи, например:

```
http://www.yoursite.com/news/items.php?item=7393
```

И теперь пусть скрипт ищет запись, имеющую id равную переменной $item, в данном случае это 7393.

Ссылка на отдельные записи

Использование URL для выбра отдельной записи можно развить, генерируя URL динамически. Это звучит немного сложно, так что расшифруем. В скрипте, работающем с базой данных бизнес-контактов, мы создадим страницу обновления, с помощью которой пользователь сможет модифицировать данные этой базы.

Для этого в таблицу вывода информации нужно добавить еще одну колонку, в ней будет размещаться ссылка на страницу обновления данной записи. Для того, чтобы указать на нужную запись, мы добавим в ссылку:

```
?id=$id
```

Выводя id записи вместе с другой информацией при выводе из базы данных, мы можем создавать ссылки, в которых указан номер ID. Затем, на странице обновления, мы просто выберем запись с этим id.

Обновление и удаление

Обновление базы данных и удаление из нее записей

Скрипт обновления базы

В предыдущем разделе мы познакомились с тем, как создавать ссылки, содержащие номер id записи, требующей обновления. Через такую ссылку мы можем передать скрипту обновления информацию о том, какая запись требует модификации. В этом случае скрипт модификации будет состоять из двух разделов.

Отображение страницы обновления

В первой части скрипта обновления используется метод выбора отдельной записи, обсуждавшийся выше, но для того, чтобы сделать его полезнее, мы добавим немного HTML. Прежде всего мы присоединяемся к базе данных и выбираем нужную запись:

```
$username="username";
$password="password";
$database="your_database";
mysql_connect(localhost,$username,$password);

$query=" SELECT * FROM contacts WHERE id='$id'";
$result=mysql_query($query);
$num=mysql_numrows($result);
mysql_close();

$i=0;
while ($i < $num) {
$first=mysql_result($result,$i,"first");
```

```
$last=mysql_result($result,$i,"last");
$phone=mysql_result($result,$i,"phone");
$mobile=mysql_result($result,$i,"mobile");
$fax=mysql_result($result,$i,"fax");
$email=mysql_result($result,$i,"email");
$web=mysql_result($result,$i,"web");

Здесь размещается код

++$i;
}
```

В этом скрипте оставлено место для кода страницы обновления. По сути, этот только HTML, форматирующий выходные данные:

```
<form action="updated.php">
<input type="hidden" name="ud_id"
value="<? echo "$id"; ?>">
Фамилия: <input type="text" value="ud_first"
value="<? echo "$first"?>"><br>
Имя: <input type="text" value="ud_last"
value="<? echo "$last"?>"><br>
Телефон: <input type="text" value="ud_phone"
value="<? echo "$phone"?>"><br>
Мобильный телефон: <input type="text" value="ud_mobile"
value="<? echo "$mobile"?>"><br>
Факс: <input type="text" value="ud_fax"
value="<? echo "$fax"?>"><br>
E-mail: <input type="text" value="ud_email"
value="<? echo "$email"?>"><br>
Веб-страница: <input type="text" value="ud_web"
value="<? echo "$web"?>"><br>
<input type="Submit" value="Обновить">
</form>
```

Как вы видите, этот код выведет стандартную форму, но вместо пустых текстовых полей, как в форме введения новой записи, в текстовых полях будет информация из соответствующих полей редактируемой записи.

Обновление базы данных

Следующим шагом наш скрипт должен действительно обновить базу данных. Это простая операция и действует с помощью нового запроса к базе данных:

```
UPDATE contacts WHERE id='$ud_id' SET first='$ud_first'
last='$ud_last' phone='$ud_phone' mobile='$ud_mobile'
fax='$ud_fax' email='$ud_email' web='$ud_web'
```

Этот запрос сообщает базе данных, что следует обновить таблицу бизнес-контактов, конкретно, запись у которой номер ID соответствует сохраненному в переменной $ud_id (которая содержит переданное через ссылку номер редактируемой записи) и модифицировать содержимое полей, присвоив им новые значения.

Этот запрос нужно встроить в простой скрипт:

```
$username="username";
$password="password";
```

```
$database="your_database";
mysql_connect(localhost,$username,$password);

$query="UPDATE contacts WHERE id='$ud_id' SET
first='$ud_first' last='$ud_last' phone='$ud_phone'
mobile='$ud_mobile' fax='$ud_fax'
email='$ud_email' web='$ud_web'";
mysql_query($query);
echo "Record Updated";
mysql_close();
```

Этот скрипт обновит базу данных и выдаст пользователю подтверждение.

Удаление записей

Заключительная часть разработки базы данных бизнес-контактов - создание страницы для удаления записей. Как и в случае страницы обновления, ей можно передавать номер id удаляемой записи через URL, например:

```
delete.php?id=9
```

Код для удаления записи аналогичен коду для обновления базы, за исключением слегка другого запроса MySQL query. Вместо запроса UPDATE нужно использовать:

```
DELETE FROM contacts WHERE id='$id'
```

Затем сюда нужно добавить код соединения с базой и подтверждения, как в случае скрипта обновления.

Циклы

Самое время упомянуть о другом использовании циклов при работе с базой данных. Также, как мы применяли циклы для вывода информации из базы данных, их можно использовать для выполнения запросов. Например, вы желаете изменить все записи в базе данных, в которых в поле "фамилия" указано Smith: вы хотите, чтобы все эти записи в поле "веб-сайт" имели адрес www.smith.com:

```
Стандартный код соединения с базой данных

$query=" SELECT * FROM contacts WHERE first='Smith'";
$result=mysql_query($query);
$num=mysql_numrows($result);

$i=0;
while ($i < $num) {
$id=mysql_result($result,$i,"id");
$query1="UPDATE contacts
SET web='http://www.smith.com' WHERE id='$id'";
mysql_query($query);
++$i;
}

mysql_close();
```

Завершение скрипта

Дополнительные замечания о работе с базой данных MySQL, заключительная версия скрипта

Экономия времени

Создавая сложные скрипты, работающие с базами данных, вы заметите, что чаще всего вы пишете код присоединения к базе данных. Поэтому вы реально сэкономите свое время, если создадите файл, в которм будет содержаться имя пользователя и пароль, или просто файл присоединения к базе. Например, вы можете создать файл dbinfo.inc.php и разместить в нем следующий код:

```
<?
$username="имя_пользователя";
$password="пароль";
$database="имя_базы_данных";
?>
```

вставив в соответствующие секции реальные значения. Теперь в своих php-файлах вы можете вставить в начале:

```
include("dbinfo.inc.php");
```

или

```
include("/full/path/to/file/dbinfo.inc.php");
```

Теперь вы можете в любом месте скрипта использовать переменные $username, $password и $database и не определять их заново каждый раз. Кроме того, если вам понадобится изменить эту информацию, например, если вы переехали на другой веб-хост, вам нужно будет изменить только один файл.

На том же принципе можно соединяться с базой данных, разместив код присоединения в отдельном файле, но при этом не нужно забывать о необходимости закрывать соединения с базой данных в каждом файле, иначе могут быть проблемы с сервером MySQL.

Поиск

Используя встроенную в MySQL функцию, можно организовать поиск по базе данных. Это делается с помощью функции LIKE:

```
SELECT * FROM имя_таблицы WHERE имя_поля LIKE '%$string%'
```

Функция LIKE дает указание базе данных выполнить поиск. Символ % означает, что вместо него могут находиться любые другие данные, в которых будет содержаться строка поиска $string. На его месте могут быть буквы, цифры и т.д.:

```
LIKE '%piano%'
```

выдаст все ряды, в которых в заданном поле имеется строка "piano".

Подобным образом, вы можете оставить только один знак %, так что вы можете задать положение строки поиска в текстовом поле, например:

```
LIKE 'piano%'
```

выдаст все ряды, в которых заданное поле начинается со строки "piano", так что строка:

```
The piano is next to the table.
```

не появится в результате запроса.

Завершение скрипта

На этой школе мы по кусочкам разрабатывали систему скриптов для работы с базой данных бизнес-контактов. Здесь вы можете скачать полный вариант получившегося набора скриптов в виде архива zip:[21] (open zip)

Заключение

С этой школы вы додлжны вынести основы совместного использования PHP и MySQL для создания веб-сайтов и программ, работающих с базами данных. Применение баз данных открывает массу новых возможностей и поможет сделать простой веб-сайт более мощным, поможет сэкономить время при его обновлении, позволит получать от пользователей обратную связь и еще многое другое.

Ресурсы по PHP

Русскоязычные ресурсы

PHP шаг за шагом[22] (open link)

Подробные уроки шаги с конкретными примерами для желающих освоить PHP. Начинается с Шага1: что такое PHP? И заканчивается Шагом20: Работа с графикой.

Клуб разработчиков PHP[23] (open link)

PHP-тусовка. Статьи, материалы по разным аспектам работы с PHP, правда не так много, как просто тусовки.

PHP в деталях[24] (open link)

Статьи и всякая интересная всячина, например, статья о применении XSL[2].

Гнездо: PHP[25] (open link)

Подборка ссылок на php ресурсы сети: каталоги php скриптов, документация, статьи, уроки.

Англоязычные ресурсы

PHP.NET[26] (open link)

Официальный сайт языка PHP. В частности:

PHP Manual[3]

21: http://xml.nsu.ru/php/contacts.zip
22: http://firststeps.narod.ru/html/php/php.html
23: http://phpclub.unet.ru/index.php3?l=ru
24: http://detail.phpclub.net/
2: http://detail.phpclub.net/soft/01-08-28/
25: http://gnezdo.webscript.ru/links/f_php/
26: http://www.php.net/

Описание функций для разных версий PHP[4]
Страница для скачивания PHP[5]

PHP Builder[27] (open link)

Большой и известный сайт по PHP.

Раздел PHP на hotscripts.com[28] (open link)

Ресурсы по PHP, в том числе и библиотека скриптов.

PHP Resource Index[29] (open link)

Обширная и хорошо структурированная библиотека скриптов.

Введение в PHP 3.0 и MSQL 2.0[30] (open link)

Учебник по созданию web приложений для начинающих с использованием баз данных на PHP 3.0 и MSQL 2.0. Содержит массу примеров.

PHP: faqts[31] (open link)

Большой сайт по PHP.

3: http://www.php.net/manual/en/
4: http://zugeschaut-und-mitgebaut.de/php/
5: http://www.php.net/downloads.php
27: http://www.phpbuilder.com/
28: http://www.hotscripts.com/PHP/
29: http://php.resourceindex.com/Complete_Scripts/
30: http://hagen.let.rug.nl/~s0367672/pm_int_e.htm
31: http://www.faqts.com/knowledge_base/index.phtml/fid/51/