

Лекция

Взаимодействие PHP и MySQL



В дистрибутив PHP входит класс `mysqli`, который содержит методы, свойства и функции для работы с СУБД MySQL.

Это необходимо для того чтобы вносить информацию в базу данных и просматривать ее содержимое.

При работе с web-интерфейсом для добавления информации в базу данных пользователю нужно просто ввести эти данные в html-форму и отправить их на сервер, а скрипт сделает все остальное.

А для просмотра содержимого таблиц достаточно просто щелкнуть по ссылке и зайти на нужную страницу.

ПОСТРОЕНИЕ ИНТЕРФЕЙСА ДЛЯ ДОБАВЛЕНИЯ ИНФОРМАЦИИ

Чтобы построить интерфейс для добавления информации в какую-либо таблицу базы данных, необходимо ее структуру (т.е. набор ее полей) отобразить в html-форму.

Для этого выполняем следующие действия:

- 1) устанавливаем соединения с БД;
- 2) осуществляем выбор рабочей БД;
- 3) получаем список полей таблицы;
- 4) отображаем поля в html-форму ;
- 5) данные, введенные в форму, записываем в базу данных.

1) устанавливаем соединения с БД;

Подключение к серверу СУБД MySQL производится при помощи конструктора реализованного в классе **mysqli**.

```
$mysqli = new mysqli('host', 'username', 'passwd', 'dbname', 'port');
```

После вызова конструктора класса **mysqli**, возвращает объект, представляющий подключение к серверу MySQL

ИЛИ

```
$mysqli = new mysqli();  
$mysqli->real_connect('host', 'username',  
    'passwd', 'dbname', 'port')
```

host — хост, к которому мы подключаемся

username - Имя пользователя

passwd - Используемый пароль

dbname - База данных для запросов

*Соединение с сервером закрывается при
завершении исполнения скрипта или с
помощью свойства close*

```
$mysqli->close();
```

ПРИМЕР соединения с базой данных на локальном сервере для пользователя student с паролем "123":

```
<?php  
$mysqli = new mysqli("localhost", " student  
", "123", "db");
```

```
/* проверка соединения */  
if (mysqli_connect_errno())  
{  
    printf("Соединение не установлено: %s\n",  
mysqli_connect_error());  
    exit();  
}
```

```
/* закрытие соединения */  
$mysqli->close();  
?>
```

string *\$mysqli->connect_errno*;

int **mysqli_connect_errno** (void)

Возвращает код ошибки последнего вызова
mysqli_connect().

string *\$mysqli->connect_error*;

string **mysqli_connect_error** (void)

Возвращает последнее сообщение об ошибке
после вызова *mysqli_connect()*.

2) осуществляем выбор рабочей БД;

```
$mysqli->query("SELECT DATABASE()")
```

Возвращает **TRUE** в случае успешного завершения или **FALSE** в случае возникновения ошибки.

Эта функция используется только для смены базы данных во время подключения. Вы можете выбрать базу данных, передав ее четвертым параметром в функции [mysqli_connect\(\)](#).


```
/* возвращаем имя текущей базы данных */
```

```
if ($result =  
$mysqli->query("SELECT DATABASE()))  
{  
    $row = $result->fetch_row();  
    printf("Установлено соединение с базой  
данных %s.\n", $row[0]);  
    $result->close();  
}
```

```
/* изменяем текущую базу данных на new_db */  
$mysqli->select_db("new_db");
```

mixed **mysqli_result::fetch_row** (void)

mixed **mysqli_fetch_row** (mysqli_result \$result)

Выбирает одну строку данных из результирующего набора и возвращает ее в виде массива.

Индексы элементов соответствуют номерам столбцов (начиная с 0).

int **printf** (string \$format [, mixed (string \$format [, mixed \$args [, mixed \$..]])

Выводит строку, отформатированную в соответствии с аргументом format.

Делаем базу данных book рабочей:

```
<?php
    $mysqli = new mysqli("localhost",
        "student", 123, "book");

    /* проверяем соединение */
    if (mysqli_connect_errno()) {
        printf("Соединение не
установлено: %s\n", mysqli_connect_error());
        exit();
    }

    $mysqli->close();
?>
```

```
mixed mysqli::query ( string $query  
    [, int $resultmode =  
    MYSQLI_STORE_RESULT ] )
```

```
mixed mysqli_query ( mysqli $link , string $query  
    [, int $resultmode = MYSQLI_STORE_RESULT ]  
    )
```

Выполняет запрос query к базе данных.

link

Только для процедурного стиля:

Идентификатор соединения, полученный с помощью mysqli_connect() Только для процедурного стиля: Идентификатор

query - текст запроса.

resultmode

или константа **MYSQLI_USE_RESULT**,
или **MYSQLI_STORE_RESULT** в
зависимости от требуемого поведения
функции. По умолчанию используется
MYSQLI_STORE_RESULT.

MYSQLI_USE_RESULT все последующие
вызовы этой функции будут возвращать
ошибку *Commands out of sync* до тех
пор, пока не будет вызвана функция
[mysqli_free_result\(\)](#)

Создание таблицы

```
if ($mysqli->query("CREATE TEMPORARY TABLE  
myCity LIKE City") === TRUE)  
{  
    printf("Таблица myCity успешно создана.\n");  
}
```

Select запросы

```
if ($result = $mysqli->query("SELECT Name FROM Cit  
y LIMIT 10"))  
{  
    printf("Select вернул %d строк.\n", $result->num_  
rows);  
}
```

Чтение большого объема данных

```
if ($result = $mysqli->query("SELECT * FROM  
M City", MYSQLI_USE_RESULT))
```

Очистка результирующего набора

```
$result->close();
```

`$result->fetch_row()` – получает текущий ряд результата в виде нумерованного массива,

`$result->fetch_assoc()` – в виде ассоциативного массива,

`$result->fetch_array()` – тип массива задается константой,

`MYSQLI_ASSOC` – ассоциативный,

`MYSQLI_NUM` – нумерованный,

`MYSQLI_BOTH` – оба,

`$result->fetch_object()` – строка результата в виде объекта.

У этой функции есть два параметра, оба необязательные:

`class_name` – имя класса, на основе которого будет создан объект,

`params` – массив параметров, которые будут переданы конструктору при создании объекта.

```
class Book
```

```
{  
  private $some1;  
  public $some2;  
  protected $id;
```

```
function __construct($param1, $param2) {  
  $this->some1 = $param1;  
  $this->some2 = $param2;
```

```
}
```

```
$book = $result->fetch_object('Book', array(1, 2));  
var_dump( $book);
```

```
class dbClass
```

```
{
```

```
    private $dbConstAll;
```

```
    private $dbHost;
```

```
    private $dbPort;
```

```
    private $dbLogin;
```

```
    private $dbPassword;
```

```
    private $dbName;
```

```
public function __construct($dbConstAll, $dbHost='localhost',  
    $dbPort='3306', $dbLogin='root', $dbPassword='',  
    $dbName='u138672863_asd')
```

```
{
```

```
    $this->dbConstAll = $dbConstAll;
```

```
    $this->dbHost      = $dbHost;
```

```
    $this->dbPort      = $dbPort;
```

```
    $this->dbLogin     = $dbLogin;
```

```
    $this->dbPassword  = $dbPassword;
```

```
    $this->dbName      = $dbName;
```

```
public function __destruct()  
{  
    $this->close();  
}
```

