1. Взаимодействие PHP с MySQL

При взаимодействии PHP и MySQL используется подход взаимодействия с СУБД посредством совокупности функций.

***mysql\_connect([server], [user], [password])*** – эта функция устанавливает соединение с сервером [server] MySQL и возвращает дескриптор соединения с базой данных, по которому все другие функции, принимающие этот дескриптор в качестве аргумента, будут однозначно определять выбранную базу данных. Вторым и третьим параметром этой функции являются имя пользователя СУБД ([user]) и его пароль ([password]).

***mysql\_close([идентификатор MySQL соединения])*** – эта функция разрывает соединения с указанным идентификатором. Возвращает *true*  если соединение было закрыто и *false* если соединение закрыть не удалось.Функция, в качестве аргумента, принимает идентификатор соединения.

***mysql\_select\_db([Имя БД], [Идентификатор соединения])*** - эта функция выбирает базу данных ([Имя БД]) для дальнейшей работы и все последующие SQL-запросы применяются к выбранной базе данных.

***mysql\_query([SQL-запрос])*** – отправляет SQL-запрос на сервер базы данных.

***mysql\_fetch\_array([идентификатор запроса]) –*** эта функция возвращает значения полей в виде ассоциативного массива, в качестве аргумента принимает идентификатор запроса *mysql\_query.*

***mysql\_result([идентификатор запроса], [номер элемента], [название столбца])*** – c помощью этой функции можно получить доступ к отдельному полю записи.

ПРИМЕР ИСПОЛЬЗОВАНИЯ ФУНКЦИЙ:

Создадим файл *config.php* со следующим кодом:

*<?php*

*$DBServer = localhost; //Адрес сервера с базой данных*

*$DBUser = “root”; //Имя пользователя*

*$DBPass = “password111” //Пароль*

*$DBName = “Database”; //Название нужной базы*

*$connect = mysql\_connect($DBServer, $DBUser, $DBPass);*

*$database = mysql\_select\_db($DBName, $connect);*

*?>*

В этой же директории создаем основной файл *process.php.* Пишем следующий код:

*<?php*

*Include “config.php”; //Подключаем файл написанный ранее*

*$query = mysql\_query(“SELECT \* FROM Students”); //Отправляем запрос на получение всех данных из таблицы “Students”*

*//Проверяем прошел ли наш запрос*

*If ($query) { //Если прошел, то выводим результаты на страницу*

*Echo “<table border=1>”;*

*Echo “<tr><td>Имя</td><td>E-mail</td></tr>”;*

*//Т.к. запрос возвращает несколько строк, применяем цикл*

*While ($result = mysql\_fetch\_array($query))*

*{*

*Echo “<tr><td>”.$result[‘name’].”</td><td>”.$result[‘email’].”</td></tr>”;*

*}*

*Echo “</table>”;*

*Else { //Если возникла ошибка во время зароса, то вывести сообщение и закрыть соединение*

*Echo “<p><b>Возникла ошибка: ”.mysql\_error().”</b></p>”;*

*If (mysql\_close($connect)) {*

*Echo “<br><p>Соединение прервано</p>”;*

*}*

*Else {*

*Echo “<br><p>Соединение не было прервано”;*

*}*

*Exit;*

*}*

*?>*

1. Работа с базами данных MySQL

Для работы с базами данных MySQL требуется cпециализированная Система управления базами данных (СУБД) и знание SQL-запросов.

**CREATE** – оператор создания.

Синтаксис: *CREATE TABLE table(`id`, `login`, `password`);* - запрос создает таблицу с названием «table» и с полями «id», «login», «password».

**INSERT**- оператор записи данных.

Синтаксис: *INSERT INTO table(`id`, `login`, `password`) VALUES (1, “mylogin”, “mypass”);* - запрос записывает в таблицу «table» значения: id – 1, login – “mylogin”, password – “mypassword”.

**SELECT** – оператор выбора.

Синтаксис: *SELECT \* FROM table* , где символ \* означает выбрать всё, FROM table – из таблицы с названием «table».

Если нужно выбрать какой-то отдельный столбец, либо элемент, после оператора SELECT нужно указать какой столбец вам нужен. Например: *SELECT login FROM table* получаем все логины находящиеся в таблице «table».

Если, к примеру, нужен логин и пароль той записи, id которой равен 1, то запрос будет выглядеть так: *SELECT login, password WHERE id=1 FROM table;*

1. Классы и объекты С++

**Создание класса***.* Прежде чем использовать свой класс, его нужно описать, все классы должны быть описаны до *main.*

**Код на С++**:

*Class ClassName*

*{*

*Public*

*Int Parametr1, Parametr2;*

*}*

**Создание объекта**

После того, как класс был описан, необходимо создать переменную этого типа (уже в *main*):

*ClassName a;*

Переменная типа *ClassName*  объявлена. Далее переменной *a* можно задать параметры:

*a.Parametr1 =3;*

*a.Parametr2 = 5;*

4. Массивы в С++

**Массив** – это область памяти, где могут последовательно храниться несколько значений одного типа.

**Объявление массива в С++.** Объявление массива происходит так же, как и объявление переменной, за исключением одного: после название массива, в квадратных скобках, указывается количество элементов массива.

Внимание нумерация массивов в С++ начинается с нуля, учтите это при работе с ними. Пример:

*Int massive[5] = {0, 1, 2, 3, 4, 5}; //Элементы массива указываются в фигурных скобках через запятую*

Для выполнения каких-либо операций с элементом массива, необходимо указывать его адрес. Пример не правильного кода:

*A = massive+5;*

Пример правильного кода:

*A = massive[5]+5; // В переменную заносим значение пятого элемента массива увеличенного на 5*

6. Установка и настройка OpenCart

**Установка.**

* На хостинге создаем базу данных и папку на ftp под файлы будущего сайта. В эту самую папку, с помощью FTP-клиента загружаем файлы дистрибутива;
* Далее, следует переименовать два файла: *config-dist.php* на *config.php* и файл *admin/config-dist.php* на *admin/config.php.* Это можно сделать с помощью того же FTP-клиента;
* Если хостинг на Linux/Unix, то на папки: image, image/cache, cache, download, а так же на файлы config.php и admin/config.php установите права доступа chmod 0755 или 0777;
* Далее заходим на свой сайт через браузер. Мы попадаем в меню установки, где нужно следовать инструкциям;
* После завершения установки, с помощью FTP-клиента удалить каталог *install*.

7, 13. Объекты С++, ООП

ООП позволяет объеденить данные и методы, относящиеся к одной сущности, и работать с ними, как с одним целым.

ООП привносит нам два ключевых понятии: **Класс** и **Объект**. **Класс** – это абстрактный тип данных. С помощью класса описывается некоторая сущность (ее характеристики и возможные действия). Например, класс может описывать студента, автомобиль и т.д. Описав класс, мы можем создать его экземпляр – **объект**. Объект – это конкретный представитель класса.

**Основные принципы ООП**

ООП основывается на нескольких базовых принципах:

**Инкапсуляция** – позволяет скрывать внутреннюю реализацию. В классе могут быть реализованы внутренние вспомогательные методы, поля, к которым доступ для пользователя необходимо запретить, тут и используется инкапсуляция.

**Наследование** – позволяет создать новый класс на базе другого. Класс, на базе которого создается новый класс, называется базовым, а базирующийся новый класс – наследником.

**Полиморфизм** – это способность объектов с одним интерфейсом иметь различную реализацию.

**Абстракция** – позволяет выделять из некоторой сущности только необходимые характеристики и методы, которые в полной мере(для поставленной задачи) описывают объект.

8. Связи таблиц MySQL

При проектировании базы данных, здравый смысл подсказывает нам, что мы должны использовать различные таблицы для разных данных. Пример: клиенты, заказы, записи, сообщения и т.д. Так же мы должны иметь взаимосвязи между этими таблицами. Например, клиент имеет заказы, а у заказов есть позиции(товары). Эти взаимосвязи должны быть отражены в базе данных. А так же, когда мы получаем данные с помощью SQL, мы должны использовать определенные типы запросов ***JOIN***,чтобы получить нужный результат.

Для организации связей между таблицами необходимо, во время создания таблиц задать внешние ключи (PRIMARY KEY – первичный ключ, FOREIGN KEY – вторичный ключ).

Создадим следующий запрос:

***CREATE TABLE*** *students(Student\_id* ***INT*** *AUTO\_INCREMENT****,***

*Name* ***VARCHAR(****100****),***

***FOREIGN KEY (****Adress\_ID****) REFERENCES*** *address(Adress\_ID) // Указываем что поле Adress\_ID является вторичным ключом аналогичного поля в таблице “address”*

*);*

Отправляем и пишем следующий запрос на создание второй таблицы:

***CREATE TABLE*** *address ( Adress\_ID* ***INT*** *AUTO\_INCREMENT* ***PRIMARY KEY, //*** *Указываем что Adress\_ID имеет в данной таблице первичный ключ*

*Name* ***VARCHAR(****100****)***

*);*

Таблицы созданы и связаны между собой, внесем данные для наглядности, как это работает.

**Результат :**

*Таблица students*

|  |  |  |
| --- | --- | --- |
| **Student\_ID** | **Name** | **Adress\_ID** |
| 12 | Сергей | 301 |

*Таблица address*

|  |  |
| --- | --- |
| **Adress\_ID** | **Name** |
| 301 | Ул. Ленина д.42 кв.54 |

Теперь поле с названием “**Adress\_ID**” из таблицы *students* будет ссылаться на соответствующую запись в таблице *address*.

Оба столбца “**Adress\_ID**” должны быть одного и того же типа. Так же следует помнить что вторичный ключ не может существовать без первичного, т.к. ему некуда будет ссылаться.

9. Bootstrap

Bootstrap – Простой и легко настраиваемый HTML, CSS и JavaScript фреймворк для более быстрой и удобной Web-разработки.

14. Ввод данных в таблицы MySQL

Для того, чтобы ввести данные в таблицу MySQL необходима СУБД с поддержкой SQL-запросов и уже с подключенной к ней базой данных (либо использовать дистрибутив PHPMyAdmin и заполнить таблицы визуально).

SQL-запрос: ***INSERT INTO*** ***`***название таблицы***`(`***столбец1***`, `***столбец2***`) VALUES (`***Значение1***`, `***Значение2***`);***

Итак, после *INTO* пишется название вашей таблицы, затем, в скобках, через запятую, название столбцов которые вы хотите заполнить, затем оператор *VALUES* и в скобках, так же через запятую, указать значения, которые будут внесены в соответствующие ячейки.

*Внимание! Значение вводятся в том порядке, в каком были введены названия столбцов. Т.е. в* ***столбец1*** *будет записано* ***Значение1****, а в* ***столбец2*** *–* ***Значение2.***

15. Сетка Bootstrap

Система разметки используются для создания макетов страниц с помощью ряда строк и столбцов, в которых размещается контент. Вот так работает система разметки Bootstrap:

* ***.row*** должны быть помещены в ***.container*** *(fixed-width)* или ***.container-fluid*** *(full-width)* для правильного выравнивания и заполнения;
* Использовать строки для создания горизонтальных групп столбцов;
* Контент должен быть помещен в столбцах, и только колонки могут быть непосредственными потомками строк;
* Стандартные классы разметки как ***.row*** и ***.col-xs-4*** доступны для быстрого принятия разметки макетов. *Less Mixins* так же может быть использован для семантических схем;
* Столбцы создают желоба (промежутки между содержанием столбцов) с помощью ***padding.*** Тогда *padding* компенсируется в строках для первого и последнего столбца с помощью отрицательного *margin* на ***.row*** ;
* Столбцы разметки создаются с указанием количетва двенадцати доступных столбцов, которые нужно охватить. Например, три равные колонки использовали бы три ***.col-xs-4***.

17. Логические операции С++

В С++ существует три логические операции:

1. Логическая операция И (***&&***);
2. Логическая операция ИЛИ (***||***);
3. Логическая операция НЕ (***!***) или логическое отрицание;

Логические операции образуют сложное (составное) условие из нескольких простых (два или более) условий. Эти операции упрощают структуру программного кода в несколько раз.

18. Обработчик событий JavaScript

Обработчик события можно указать в виде атрибута html-тэга.

Например, для обработки события *Click(клик по элементу)* на кнопке *Input,* можно назначить обработчик *onclick* вот так:

*<input id =”b1” value=”Нажми меня” onclick=”alert(‘Спасибо!’);” type=button />*

События и их обработчики:

***Onclick*** – возникает когда был произведен щелчок левой кнопкой мыши по элементу

***Oncontextmenu*** – возникает когда был произведен щелчок правой кнопкой мыши по элементу

***Onmouseover*** - возникает когда на элемент наводится мышь

***Onmousemove*** – возникает когда мышь находится в движении

***Onmousedown*** – возникает когда кнопку мыши нажали

***Onmouseup*** – возникает когда кнопку мыши отжали

***Onkeydown*** – возникает когда посетитель нажимает клавишу

***Onkeyup*** – возникает когда посетитель отпускает клавишу