Методические указания для выполнения
лабораторной работы № 6 по курсу «Web-технологии»

Серверные сценарии. Работа с базой данных.

Цель работы: изучение трехуровневой архитектуры веб-приложений (Клиент – Веб-сервер – Сервер БД) и языка написания серверного сценария php для построения динамического контента web-сайтов.

Правила оценивания работ

Выполнение базового задания оценивается максимально в 6 баллов. Для получения за работу 10 баллов требуется выполнить не менее пяти пунктов из раздела «Расширенное задание».

Из максимально возможной оценки баллы будут вычитаться при следующих ситуациях:

1) **Срок.** Отчет выложен в лк после срока сдачи – минус 1 балл каждые 2 недели от даты, записанной в ЛК. Дата защиты не влияет;
2) **Защита.** «Пробелы» в знаниях во время устной защиты работы: оценка «хорошо» - минус 1 балл, оценка «удовлетворительно» - минус 2 балла;
3) **Отчет.** В отчете присутствуют незначительные нарушения требований - минус 1 балл, отсутствует несколько пунктов отчета - минус 2 балла или больше;
4) **Работа.** В выполненном задании отсутствуют второстепенные элементы - минус 1 балл, отсутствует несколько пунктов задания - минус 2 балла или больше.

Защита работы возможна только при наличии в ЛК отчета.

Базовое задание

Установить компоненты 3-х уровневой архитектуры. В качестве звеньев 3-х уровневой архитектуры разрешается использовать Apache+PHP+MySQL (пакет программ WAMP – XAMPP, Денвер и т.п.). Язык серверных сценариев - PHP.

Подготовить реляционную базу данных, состоящую из 2 таблиц, связанных между собой отношением «один-ко-многим». Каждая таблица должна находиться в 3 нормальной форме и содержать не менее 5 полей. Содержимое полей таблицы должно соответствовать теме сайта. Запрещается делать таблицы «книга», «автор», «статья» и их аналоги. Все данные из таблиц должны отображаться на странице, включенной в разрабатываемый сайт. Для формирования веб-страницы использовать команду echo.

Расширенное задание

1.. не отображать служебные поля (первичные и вторичные ключи).
2.. все поля отобразить в одной таблице на веб-странице
3.. добавить возможность ввода новой строки в таблицу базы на веб-странице
4.. добавить возможность редактировать строку в таблицу базы на веб-странице
5.. добавить возможность удаления строки из таблицы базы на веб-странице
6.. написать серверный сценарий на php, выполняющий задание с матрицами из работы № 3.
7.. применить оформление через css
8.. несколько раз для формирования веб-страницы использовать функцию printf()

Содержание отчета

1) титульный лист;
2) цель работы;
3) вариант задания;
4) структура таблиц БД;
5) доказательство нахождения таблиц в 3-ей нормальной форме
6) данные в таблицах
7) HTML код
8) код серверных скриптов, используемых на сайте;
9) скриншоты страниц сайта

Технологии PHP, Apache, MySQL.

PHP –серверный язык сценариев. Он позволяет реализовать различные функции для веб, например: построение/поиск/сохранение данных сайта в таблицах БД, динамическое построение HTML-страниц, автоматическую рассылку почты и т.п. PHP поддерживает работу со многими СУБД, в том числе MySQL.

Пример php-скрипта:

```
<html>  <head>
<title>Example</title>  </head>
<body>
<?php echo "Hi, I'm a PHP script!"; ?>
</body>  </html>
```

php-скрипт встраивается в html-страницу при помощи специальных тегов

<?php – открывающий тег
?> - закрывающий тег

При этом расширение файла меняется **с *.html* на *.php***.
Скрипт дополняет текст веб-страницы, внутри которой он размещен. Эти дополнения оформляются как вывод на экран через команду echo:

```
echo "Вы добавили цветок: <strong>".$_POST['flower']."</strong><br />";
```

Строки заключаются в двойные или одинарные кавычки. Вид кавычек задает способ отображения строки. Внутри строки могут находиться теги html и имена переменных ($имя). Если строка заключена в **двойные кавычки**, то вместо имени переменной **подставляется ее значение**.
Операция склейки строк обозначается точкой. Можно склеивать не только строки, но и данные других типов.

Также для формирования веб-страницы применяется функция printf(), аналогичная одноименной функции в языке C:

```
printf("%s (%s)\n", $row['Name'], $row['Population']);
```

PHP выполняется на стороне веб-сервере. Т.е. для выполнения скрипта *add_flower.php* недостаточно ввести в браузере file://C:/mysite/www/add_flower.php
Нужно использовать обращение к веб-серверу: http://mysite.ru/add_flower.php

Перед каждой переменной в PHP ставится символа доллара $. Объявление переменной выполняется через оператор присваивания. Применяется динамическая типизация (тип данных переменной определяется по типу присваиваемого значения).
Пример: $my_name = “John”;

Работа с HTML-формами при помощи PHP

POST, GET запросы отправляют данные от клиента (html-страницы) к серверу (PHP-скрипту). GET – обычно используется для поиска данных (без изменения данных на сервере, имеет ограниченное кол-во параметров)
POST – используется для модификации данных (имеет неограниченное кол-во параметров)

Для передачи данных с HTML-страницы к PHP-скрипту используют:

1 Тег *form* и его атрибуты *action*, *method*
<FORM ACTION="email.php" METHOD="POST">
Тег задает экранную форму, в которой будут указываться данные для передачи. *Action* – PHP-скрипт, которому будут передаваться данные с HTML-страницы. *Method* – тип запроса (GET или POST)

2 Теги *input, select, textarea* и их атрибуты *name*
<INPUT TYPE="text" NAME="flowerColor" value="blue">
Задают данные для передачи. В данном случае будет передаваться параметр “*flowerColor*”, значение параметра: “*blue*”
3. тег *input* типа *submit*:
<INPUT TYPE="submit" VALUE="Отправить запрос!">
Используется для отправки данных экранной формы к PHP-скрипту.

Пример HTML-страницы *example.html*
Содержит экранную форму, которая передает скрипту *add_flower.php* методом *POST* два параметра – *flower* и *color*:

```
<html>
<head>
<title>Example</title>
</head>
<body>
<form action="add_flower.php" method="post">
<input type="text" name="flower" value="роза">
<input type="text" name="color" value="красный">
<input type="submit" value="Отправить!">
</form>
</body>
</html>
```

В php-скрипте переданные данные из html-форм хранятся в ассоциативных массивах *$_GET* и *$_POST* соответственно. Для обращения к элементу используется название переданного параметра: $_POST['color'].

Доступ к переданным значениям – по названию атрибута *name* тегов *input, select, textarea*
Во время отладки программы посмотреть содержимое этих массивов можно при помощи функции *<?php phpinfo(32); ?>*
Пример: файл *add_flower.php*

```
<html>
<body>
<p>
<?php
/* Этот скрипт использует значения из массива $_POST */
echo "Вы добавили цветок: <strong>".$_POST['flower']."</strong><br />";
echo "Его цвет: <strong>".$_POST['color']."</strong>"; ?>
</p>
</body>
</html>
```

На экране:
Вы добавили цветок: **роза**.
Его цвет: **красный**.

Операторы php

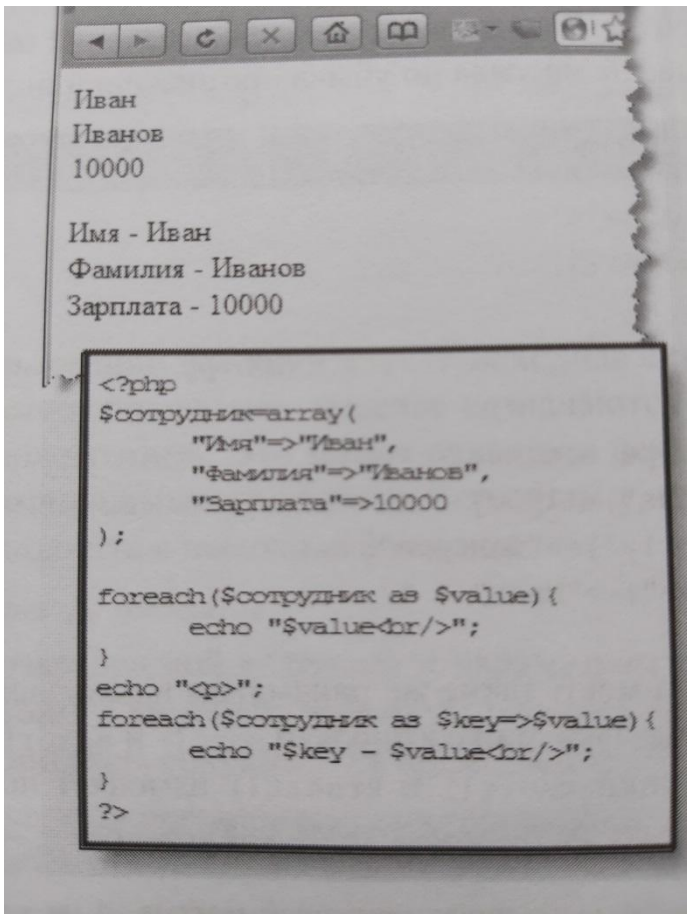
Большая часть операторов повторяет операторы языка С.
Для перебора элементов массива введен специальный цикл foreach, рассмотрим его синтаксис. Существует два варианта написания:

```
foreach ( массив  as  $значение)
{}

foreach ( массив  as  $индекс => $значение)
{}
```

Здесь $значение – имя переменной, в которой находятся значения элементов массива. А $индекс – переменная, содержащая индекс (ключ) элемента массива.

```
foreach($_POST as $key=>$value)
 echo "$key=$value <br/>";
```



Работа с пакетом XAMPP.

Существует достаточно много различных пакетов с набором инструментов для имитации работы веб-сервера (wamp- пакеты). Пакет XAMPP работает стабильно, однако его легко можно заменить на какой-либо другой. Пакет бесплатный, устанавливается быстро.

Перед началом работы с веб-сервером и базой данных, необходимо запустить сервер Appache и СУБД MySQL (кнопки start на панели управления).

Чтобы создать свой сайт в пакете XAMPP, необходимо в папке C:\xampp\htdocs создать папку, например wordpress, чтобы в дальнейшем можно было обращаться к данному сайту по адресу http://localhost/wordpress/ Для этого на панели управления XAMPP надо нажать кнопку Explorer, которая открывает каталог пакета.

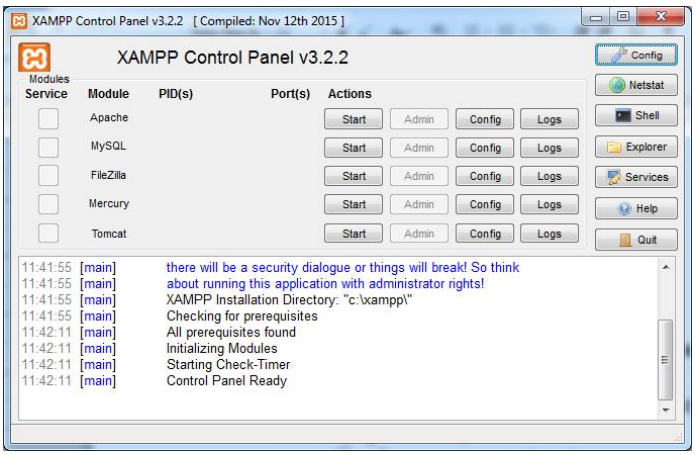


Рис. 1 Окно пакета XAMPP

Для работы с базой данных через окно phpMyAdmin в адресной строке браузера при запущенном пакете XAMPP надо набрать localhost. Появится окно, показанное на рис. 2. В этом окне нажать кнопку phpMyAdmin, после чего появится окно, изображенное на рис. 3.

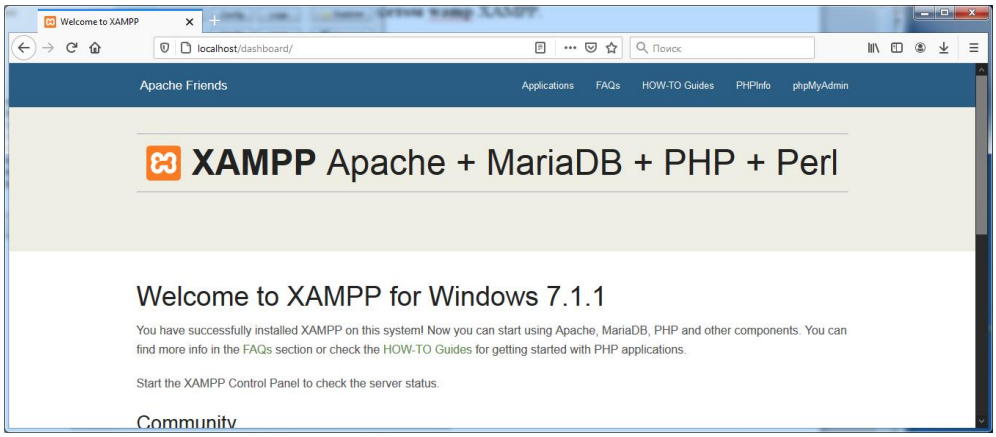


Рис. 2. Окно пакета XAMPP в браузере

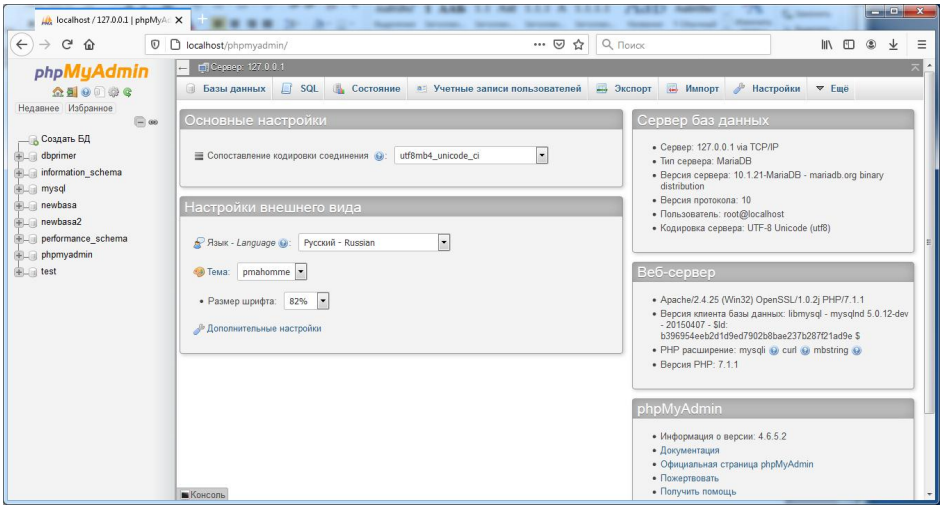


Рис. 3 Окно СУБД в браузере

Для доступа к содержимому таблиц базы данных используется язык SQL (структурированный язык запросов). Пример запроса к таблице *MY_FLOWERS*:

*SELECT * FROM MY_FLOWERS WHERE COLOR = “Blue”*

Пример кода на PHP для вывода на экран содержимого таблицы БД

/* создать соединение */
mysql_connect($hostname,$username,$password) OR DIE("Не могу создать соединение ");

/* выбрать базу данных. Если произойдет ошибка - вывести ее */ mysql_select_db($dbName) or die(mysql_error());

/* составить запрос, который выберет всех клиентов - яблочников */
$query = "SELECT * FROM $userstable WHERE choise = 'Яблоки'";

/* Выполнить запрос. Если произойдет ошибка - вывести ее. */
$res = mysql_query($query) or die(mysql_error());

/* Как много нашлось таких */
$number = mysql_num_rows($res);

/* Напечатать всех в красивом виде*/
if ($number == 0) { echo "<CENTER><P>Любителей яблок нет</CENTER>"; } else { echo "<CENTER><P>Количество любителей яблок: $number

";

/* Получать по одной строке из таблицы в массив $row, пока строки не кончатся */
while ($row=mysql_fetch_array($res))
{ echo "Клиент ".$row['name']." любит Яблоки.
"; echo "Его Email: ".$row['email']; }
echo "</CENTER>"; }

Применение функций mysqli.

Сейчас для доступа к базе данных рекомендуется использовать обновленный вариант функций. К их названию добавляется буква i. Общая логика повторяет более старый вариант, но есть и отличия. Вот простой скрипт, который соединяется с сервером MySQL, посылает запрос серверу с помощью этого соединения, выводит результаты запроса и затем освобождает результирующее множество запроса и закрывает соединение.

```
<?php

/* Подключение к серверу MySQL */
$link = mysqli_connect(
            'localhost',  /* Хост, к которому мы подключаемся */
            'user',       /* Имя пользователя */
            'password',   /* Используемый пароль */
            'world');     /* База данных для запросов по умолчанию */

if (!$link) {
   printf("Невозможно подключиться к базе данных. Код ошибки: %s\n",
mysqli_connect_error());
   exit;
}
```

```
/* Посылаем запрос серверу */
if ($result = mysqli_query($link, 'SELECT Name, Population FROM City ORDER BY
Population DESC LIMIT 5')) {

    print("Очень крупные города:\n");

    /* Выборка результатов запроса */
    while( $row = mysqli_fetch_assoc($result) ){
        printf("%s (%s)\n", $row['Name'], $row['Population']);
    }

    /* Освобождаем используемую память */
    mysqli_free_result($result);
}

/* Закрываем соединение */
mysqli_close($link);
?>
```

Приведенный сценарий должен вывести такие данные:

```
Очень крупные города:

Mumbai (Bombay) (10500000)
Seoul (9981619)
Sao Paulo (9968485)
Shanghai (9696300)
Jakarta (9604900)
```