МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра информационной безопасности

РАЗРАБОТКА И ЗАЩИТА WEB-ПРИЛОЖЕНИЙ С СЕРВЕРНЫМИ СЦЕНАРИЯМИ НА ЯЗЫКЕ PHP

Методические указания по выполнению лабораторных работ
для студентов направления подготовки (специальности)
09.03.02 Информационные системы и технологии

Курск 2021

УДК 004.56.5(076.5)
Составитель: А.Л. Ханис

Рецензент
Кандидат технических наук, доцент кафедры
информационной безопасности А.Л. Марухленко

Разработка и защита Web-приложений с серверными сценариями на языке PHP : методические указания по выполнению лабораторных работ студентов всех форм обучения / Юго-Зап. гос. ун-т; сост.: А.Л. Ханис. - Курск, 2021. - 33 с.: ил. 2, табл. 1. – Библиогр.: с. 33.

Содержат краткие теоретические положения об основных операторах языка программирования PHP, а также о реализации авторизации и регистрации пользователей с помощью PHP-скриптов.

Методические указания соответствуют требованиям программы по направлению подготовки бакалавров: математическое обеспечение и администрирование информационных систем.

Предназначены для студентов направления подготовки бакалавров 09.03.02.

Текст печатается в авторской редакции

Подписано в печать . Формат 60x84 1/16.
Усл.печ. л. 1,86 Уч.-изд. л. 1,68. Тираж 100 экз. Заказ. Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

ВВЕДЕНИЕ

Официальное название языка PHP – Hypertext Processor (гипертекстовый препроцессор). Он является языком сценариев, выполняющихся на сервере. Когда браузер пытается получить доступ к указанному URL, по которому расположена PHP-страница, он выполняет сначала запрос к веб-серверу, который активизирует синтаксический анализатор PHP. Затем этот анализатор выполняет PHP-код, расположенный в запрашиваемом файле, и возвращает полученный результат веб-серверу как часть HTML-кода. Данный HTML-код, в свою очередь, передается в браузер пользователя для отображения. Именно этот результат и видит пользователь, когда запрашивает PHP-страницу. Следует отметить, что при этом пользователь не имеет возможности просмотреть сам PHP-код, в отличие от JavaScript, а также не требуется наличия специальных интерпретаторов в браузере.

Согласно руководству по PHP «Цель языка – помочь веб-разработчику быстро создавать динамически генерируемые страницы».

Обычно PHP используется для следующих целей:

- выполнение системных функций – создание, открытие, чтение, запись и закрытие файлов, выполнение системных команд, создание каталогов, изменение прав доступа;
- сбор данных с помощью форм – сохранение данных в файле, отправка данных по e-mail, возврат обработанных данных пользователю;
- доступ к базам данных и генерация содержимого «на лету» (технология отображения данных в зависимости от запроса пользователя) или создание веб-интерфейса для добавления, удаления и изменения элементов в базе данных;
- запись на компьютер пользователя файлов «cookie» и доступа к переменным«cookie»;
- запуск сессий и использование переменных объектов сессии;
- проверка имени и пароля (аутентификация) пользователя для ограничения его доступа к разделам веб-сайта;
- шифрование данных;

- организация электронных платежей и др.

Следует отметить межплатформенность PHP (он способен работать на различных платформах и с различными веб-серверами). При своей многофункциональности, стабильности и надежности в работе PHP достаточно нетребователен к системным ресурсам веб-сервера. Также он не требует приобретения лицензии на использование, что делает его особенно привлекательным как для коммерческих, так и для личных целей.

Основные преимущества PHP проявляются при совместном использовании с базой данных MySQL. В этом случае вы можете раскрыть полный потенциал динамических Web-страниц и перейти к созданию системы Web-публикации, в основе которого лежит именно широкое использование баз данных. На сегодняшний день MySQL является лидером среди систем управления базами данных для разработки интернет-приложений. Область применения MySQL самая разнообразная – от создания форумов, гостевых книг, досок объявлений до интернет-магазинов и корпоративных веб-сайтов. Если веб-проект содержит много информации, то в нем не обойтись без базы данных. Именно база данных позволяет структурировать информацию и обеспечивает оперативный и бесперебойный доступ к данным. Учитывая возможности, стабильность, надежность, безопасность и распространенность MySQL, выбор в пользу этой базы данных становится очевидным.

SQL (Structured Query Language) – структурированный язык запросов. SQL создан для работы с реляционными базами данных. Он позволяет пользователям взаимодействовать с базами данных (просматривать, искать, добавлять и удалять данные). Таблицы в реляционной базе данных связаны между собой при помощи отношений. Поэтому при выполнении запроса обеспечивается возможность объединить данные из нескольких таблиц. SQL как часть системы MySQL можно охарактеризовать как язык структурированных запросов плюс наиболее распространенный язык, используемый для доступа к базам данных.

Лабораторная работа

Разработка и защита Web-приложений с серверными сценариями на языке PHP

1. Установка и настройка веб-сервера с PHP

Цель: научиться устанавливать на персональный компьютер программное обеспечение, необходимое для программирования и отладки веб-сценариев.

Задание: установить на flash-накопитель веб-сервер Apache, интерпретатор языка PHP, текстовый редактор Notepad++, веб-браузер Firefox с расширением Firebug.

Ход работы.

Для изучения дисциплины и полноценной работы необходимо установить веб-сервер Apache (http://apache.org/) и интерпретатор языка программирования PHP (http://php.net/). Установка этого ПО достаточно сложное дело для начинающих пользователей (как это делается и для чего это необходимо можно прочитать здесь http://web.diwaxx.ru/web-server-doma.php), поэтому мы будем использовать готовую сборку Denwer (http://www.denwer.ru/). Инструкция по установке находится по адресу http://www.denwer.ru/base.html. Устанавливать Denwer необходимо на flash-накопитель, чтобы была возможность программировать как на занятиях, так и в домашних условиях. Устанавливайте Denwer в каталог первого уровня flash-накопителя, например f:\WebServers. Под конец установки будет задан вопрос, как именно Вы собираетесь запускать и останавливать комплекс. Есть два альтернативных варианта:

1. Создавать виртуальный диск при загрузке машины (при этом, инсталлятор обеспечит, чтобы это происходило автоматически), а при остановке серверов виртуальный диск не отключать.

2. Создавать виртуальный диск только по явной команде старта комплекса. И, соответственно, отключать диск от системы — при остановке серверов.

Первый режим наиболее удобен, если комплекс

устанавливается на жесткий диск

компьютера, а не на flash-накопитель, поэтому необходимо выбрать второй вариант. На вопрос создавать ярлыки для запуска и остановки Denwer на рабочем столе отвечайте отрицательно. Для запуска комплекса будем использовать файлы Run.exe, Stop.exe и Restart.exe в каталоге X:\WebServers\denwer\, где X:\WebServers\ - диск и папка, в которую установлен комплекс.

При запуске комплекса брандмауэр операционной системы может заблокировать запуск веб-сервера Apache. В этом случае брандмауэр потребует подтверждения Ваших намерений. Щелкните на Don't Block Anymore (не блокировать в дальнейшем).

Возможно, что на том компьютере, где Вы захотите запустить Denwer с Вашего flash-накопителя, Denwer не сможет создать виртуальный диск с той буквой, которую Вы задали при установке комплекса, т.к. в системе уже будет такой диск. В этом случае необходимо поменять букву виртуального диска, изменив в файле конфигурации X:\WebServers\denwer\CONFIGURATION.txt строку:

subst_drive = Z:

где необходимо поменять букву Z на любую другую, для которой в системе нет диска. Ни в коем случае не меняйте ничего больше в файле конфигурации, это может привести к нарушению работы комплекса.

Кроме Denwer Вам также потребуется текстовый редактор (не путать с текстовым процессором, подобным MS Word), желательно с подсветкой синтаксиса языков программирования и разметки, и современный веб-браузер с инструментами отладки. В качестве текстового редактора можно использовать Notepad++ (http://notepad-plus-plus.org/), а в качестве веб-браузера — Firefox с расширением Firebug. Их также необходимо установить на flash-накопитель.

Для проверки работы Denwer запустите файл X:\WebServers\denwer\Run.exe и наберите в браузере http://localhost. Если по этому адресу откроется служебная страница

Denwer, значит все работает исправно, иначе идем на

страницу http://www.denwer.ru/base.html и пытаемся разобраться с настройками сети и прокси-сервером.

Для остановки Denwer выполните файл X:\WebServers\denwer\Stop.exe.

1. Вывод сообщения в языке PHP

Задания:

1. Откройте текстовый редактор (или Dreamweaver) и наберите следующий код:

Листинг № 1.

```
<html>
<head><title>Вставка кода PHP</title></head>
<body>
<h1>Пример страницы с PHP кодом</h1>
<?
print(“<h2> PHP-фрагмент </h2>”);
?>
</body>
</html>
```

Или

Листинг 2

```
<?
print(“<html><head><title>Вставка кода
PHP</title></head><body>”);
print(“<h1>Пример страницы с PHP кодом</h1>”);
print(“<h2> PHP-фрагмент </h2>”);
print(“</body></html>”);
?>
```

После этого выполните следующие действия:

1. Сохраните данный текстовый файл в каталоге **C:\WebServers\home\localhost\www\xp** (каталог **xp** необходимо создать) под именем **php_start1.php** (обратите внимание, расширение у файла .php).

2. Запустите web-сервер **Apache** и в строке адреса браузера наберите **http://localhost/xp**. Вы должны увидеть внутри виртуального каталога **xp** свой файл **php_start.php**.

3. Щелкните на нем и, если вы правильно набрали приведенный код, у вас должна загрузиться страница (рис 1.).

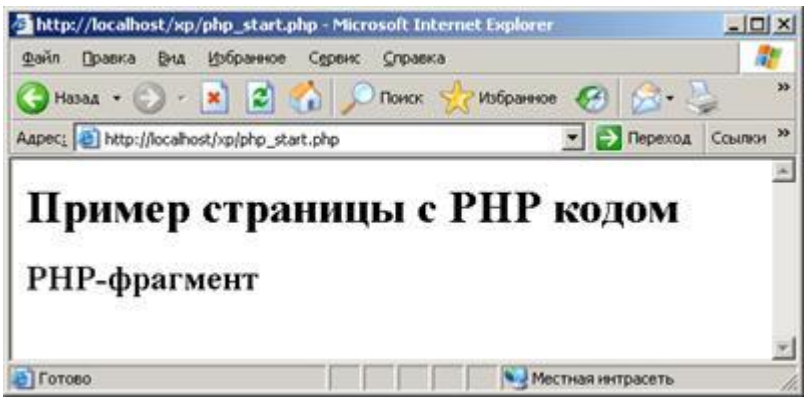

Рис. 1 – Web-страница программы «вывод сообщения»

4. Посмотрите html-код данной страницы (меню Вид → Просмотр HTML кода).

1. Передача переменных

1. Создайте новый html-файл (forma.html), содержащий следующий код (листинг 12):

Листинг №3.

```
<html>
<head>
<title>ввод значений в форму</title>
</head>
<body>
<form action="age.php" method="get">
```

```
<p>ваше имя: <input name="user_name" size="20" type="text"></p>
<p>год рождения: <input name="user_yare" size="20" type="text"></p>
<input name="b1" type="submit" value="отправить"><input name="b2" type="reset" value="очистить">
</form>
</body>
</html>
```

2. Создайте файл **age.php** и наберите в нем код, представленный в листинге №4.

Листинг №4.

```
<html>
<head>
<title>вычисление возраста</title>
</head>
<body>
<p>Добро пожаловать <? echo ($user_name) ?></p>
<?
$yare = date("Y");
$user_age = $yare - $user_yare;
print ("<p>вам  $user_age лет</p>");
?>
</body>
</html>
```

echo(); - функция отвечающая за вывод значений переменных, во многом, аналогична функции **print().**

date(); - функция возвращающая текущую дату в виде строки. Функция имеет большое количество параметров.

Параметры функции date():

G - час, 24-часовой формат без ведущих нулей; т.е. от «0" до «23"

i - минуты; т.е. от «00" до «59"
j - день (число) месяца без ведущих нулей; т.е. от «1" до «31"
m - месяц; т.е. от «01" до «12"
H - час, 24-часовой формат; т.е. от «00" до «23"
n - месяц без ведущих нулей; т.е. от «1" до «12"
s - секунды; т.е. от «00" до «59"
Y - год, 4 цифры; например, «1999"
y - год, 2 цифры; например, «99"
z - день года; т.е. от «0" до «365"

пример использования функции: date()
$today = date("j, n, Y"); переменная $today примет значение: 10, 3, 2001 (число, месяц, год)
$today = date("H:i:s"); переменная $today примет значение: 17:16:54 (часы, минуты, секунды)

3. Введем имя пользователя, например, **Dik**, а год его рождения **1973,** нажав кнопку «отправить», данные будут переданы файлу **age.php**, и строка адреса примет вид:

http://localhost/xp/age.php?user_name=Dik&user_yare=1973&b1=%EE%F2%EF%F0%E0%E2%E8%F2%FC

– все передаваемые данные располагаются за символом **“?”**
– все данные собраны в виде: имя переменной = значение переменной;
– переменная b1 имеет значение «отправить», текст, содержащий кириллицу. Для таких переменных браузер автоматически выполняет URL-кодирование.

1. Самостоятельно создайте форму, в которой вводится имя студента, и его год его рождения. Данные из формы должны передаваться в php-файл, который определяет, на каком курсе учится человек (предполагая, что студент поступил в колледж в 16 лет).

2. Алгоритмические конструкции на PHP

Применение **условного оператора** аналогично использованию в других языках программирования. Синтаксис полностью идентичен языку Си.

```
if ($a > $b) {
   print «a больше чем b";
} else {
   print «a не больше чем b";
}
```

Возможно использование сокращенного условного оператора без оператора else, также возможно любое вложение условных операторов:

```
if ($a > $b) {
   print «a больше чем b";
} elseif ($a == $b) {
   print «a равно b";
} else {
   print «a меньше чем b";
}
```

В качестве условия могут выступать стандартные логические операции:

> (больше)
< (меньше)
= = (проверка равенства)
!= (не равно)

Также возможны логические связки:

логическое сложение - $a || $b (возможно **$a or $b**). Выражение будет истинным если хотя бы одно логическое выражение истинно.

логическое умножение - $a && $b (возможно **$a and $b**). Выражение будет истинным только если оба логических выражения истинны.

логическое умножение **- ! $a.** Истинность выражения принимает противоположное значение.

В языке PHP существует функция **isset(),** которая очень активно применяется вместе с условным оператором. Ее назначение проверять наличие переменной. Например, мы создаем страницу, которая проверяет текущую дату с днем рождения зарегистрированного пользователя (которое хранится в переменной $data_r) и в случае совпадения, поздравляет его с праздником. Но выполнять операцию сравнения можно лишь в случае, если в данную страницу передана переменная $data_r.

1. Создайте файл data_r.php и наберите в нем следующий программный код (листинг5):

Листинг №.5

```
<html>
<head><title>Проверка условия в языке PHP</title></head>
<body>
<H1>проверка даты рождения</H1>
<form action="data_r.php" method="get">
<P>месяц рождения: <input name="m" size="5" type="text"></P>
<P>день рождения: <input name="d" size="5"  type="text"></P>
<P><input name="b" type="submit" value="проверить"></P>
</form>
<?
if(isset($b)){
$day = date("j");
$month = date("n");
if ($m = = $month && $d = = $day){
print("<h2>С днем рождения</h2>");
}
}
?>
</body>
```

```
</html>
```

3. Циклические конструкции

Смысл циклических конструкции в языке PHP такой же, как и в других языках программирования. Синтаксис полностью идентичен языку Си.

Цикл «Пока»

```
$i = 1;
while ($i <= 10) {
   print $i++;
}
```

Цикл «С параметром»

```
for ($i = 1; $i <= 10; $i++) {
   print $i;
}
```

4. Практические задания:

1. Для изучения условного оператора создайте тест, состоящий из пяти вопросов (файл **test.htm**). После выбора правильных ответов, данные передаются в новый файл **analyse_test.php**, где вычисляется количество правильных ответов и выводится соответствующее сообщение. Обратите внимание, при ответе пользователь мог специально или случайно пропустить вопрос, поэтому перед проверкой каждого ответа на правильность нужно проверить, а передана ли соответствующая переменная в php-файл.

2. Создайте PHP-страницу, в которой пользователь вводит в форму количество строк и количество столбцов. В результате по введенным значениям строится таблица. Примерный вид экрана представлен на рисунке 2.

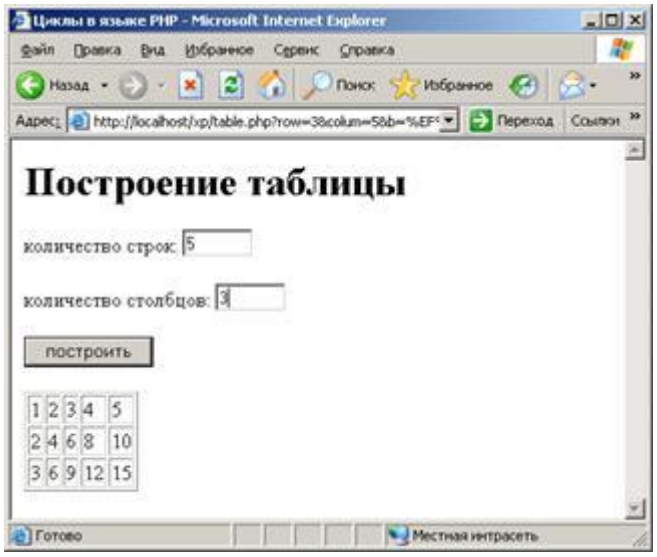

Рис. 2 – Web-страница программы «построение таблицы»

Работа с сессиями. Реальная авторизация и регистрация.

Цель занятия: научиться работать с сессиями.

Задание:
Создание авторизации и регистрации.

Ход работы.

Веб-сервер не поддерживает постоянного соединения с клиентом, и каждый запрос обрабатывается, как новый, безо всякой связи с предыдущими. То есть, нельзя ни отследить запросы от одного и того же посетителя, ни сохранить для него переменные между просмотрами отдельных страниц. Для решения этих двух задач и были изобретены сессии.

Собственно, сессии, если в двух словах - это механизм, позволяющий однозначно идентифицировать браузер и создающий для этого браузера файл на сервере, в котором хранятся переменные сеанса.

Теперь создайте файл registration.php, который будет содержать форму регистрации пользователя.

```
<form method="POST" action="registration.php">
<p>Имя пользователя: <input type="text" name="username"></p>
<p>Email: <input type="text" name="email"></p>
<p>Пароль: <input type="password" name="password"></p>
<p>Пароль(повторить): <input type="password"
name="password2"></p>
<p><input type="submit"></p>
</form>
```

В самом начале файла необходимо вызвать функцию session_start(), она создает сессию или продолжает текущую на основе текущего идентификатора сессии, который передается через запросы, такие как GET, POST или cookie. В большинстве случаев используют сессии на cookie, поэтому перед функцией start_session() не должно быть функций, возвращающих сообщение в браузер. Затем делаем проверку, аутентифицирован ли пользователь.

```
if ($_SESSION['username']) {

echo 'Вы уже зарегистрированы';
return 0;
}
```

Далее проверяем, существует ли POST запрос, если да, то проверяем совпадают ли пароли и заносим в переменную сессии данные из POST запроса.

```
if ($_POST) {

if ($_POST['password'] == $_POST['password2']) {
$_SESSION['username'] = $_POST['username'];
echo 'Пользователь '.$_SESSION['username'].' зарегистрирован';
return 0;
} else {
echo 'Введенные пароли не совпадают';
```

```
return 0;
}
}
```

В index.php заменяем все POST на SESSION.

Теперь необходимо создать файл logout.php. Этот файл будет содержать функцию, которая разрушит все данные, зарегистрированные в сессии – session_destroy().

```
<?php
session_start();

session_destroy();
header('Location: index.php');
?>
```

Перед тем как отправить форму, ее нужно проверить. Проверка формы будет осуществляться посредством javascript. Для этого у формы определим событие onsubmit="return checkForm(this)". Функция checkForm будет вызываться перед отправкой данных. Проверять будем имя пользователя, заполнено он или нет и email.

Шаблон email`a будет [английские_буквы]@ [английские_буквы]. [английские_буквы]. Такую проверку можно сделать, используя регулярные выражения - это формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании. По сути это строка-образец, состоящая из символов и метасимволов и задающая правило поиска. Более подробно

о регулярных выражениях можно почитать на Википедии (http://ru.wikipedia.org/wiki/Регулярные_выражения).

Вообще регулярное выражение для проверки электронной почты довольно громоздкое и сложное для понимания, поэтому воспользуемся простой формой [a-zA-Z]*@[a-zA-Z]*\.[a-zA-Z].

В начале функции определим все переменные, которые нам понадобятся, и массивы с ошибками.

```
var el, // Сам элемент
elName, // Имя элемента формы
value, // Значение
type; // Атрибут type для input-ов
reg = /[a-zA-Z]*@[a-zA-Z]*\.[a-zA-Z]/;
var errorList = [];

var errorText = {

1      : "Не заполнено поле 'Имя'",
2      : "Не заполнено поле 'E-mail'",
}
```

Далее проходимо по всем элементам формы и проверяем все теги input. Если поля для имени пустое или электронная почта не соответствует шаблону, записываем номера ошибок в массив.

```
for (var i = 0; i < form.elements.length; i++) { el = form.elements[i];
elName = el.nodeName.toLowerCase();

value = el.value;
if (elName == "input") {
type = el.type.toLowerCase();
switch (type) {
case "text" :

if (el.name == "name" && value== "")
errorList.push(1);
if (el.name == "email" && !value.match(reg))
errorList.push(2);
break;

default :
```

```
break;
}
}
}
```

Затем проверяем массив с ошибками. Если он пуст, то возвращаем true, иначе формируем текст для ошибки, выводм это на экран и возвращаем false.

```
if (!errorList.length) return true;

var errorMsg = "При заполнении формы допущены следующие ошибки:\n\n"; for (i = 0; i < errorList.length; i++) {
errorMsg += errorText[errorList[i]] + "\n";
}
alert(errorMsg);
return false;
```

Чтение и запись в файл. Регистрация с записью в файл. Авторизация из файла.

Цель занятия: научиться работать файлами.

Задание:
Создание авторизации и регистрации на файлах.

Ход работы.
Для работы с файлами в php существует несколько функций.
fopen – функция для открытия файла.
$fp = fopen('filename, 'param');

filename и param это обязательные параметры. Первый отвечает за имя файла, который необходимо открыть, а второй определяет режим файла:

1. r – открытие файла только для чтения.
2. r+ - открытие файла одновременно на чтение и запись.

3. w – создание нового пустого файла. Если на момент вызова уже существует такой файл, то он уничтожается.

4. w+ - аналогичен r+, только если на момент вызова фай такой существует, его содержимое удаляется.

5. a – открывает существующий файл в режиме записи, при этом указатель сдвигается на последний байт файла (на конец файла).

6. a+ - открывает файл в режиме чтения и записи при этом указатель сдвигается на последний байт файла (на конец файла). Содержимое файла не удаляется.

Записывать данные в файл при помощи PHP можно при помощи функции fwrite(). Это функция принимает 2 обязательных параметра и 1 необязательный. В качестве обязательных параметров выступает дескриптор файла и режим файла:

```
$test = fwrite($fp, $mytext);
```

По завершению работы с файлом, его нужно закрыть, используя функцию fclose($fp).

Теперь в файле registration.php после строки if '($_POST) {' поставим проверку файла, если он существует, то открываем его и перемещаем указатель в конец строки, если нет, то создаем его.

```
$fp = fopen('users.txt', 'a+');
if (!$fp) {
$fp = fopen('users.txt', 'w+');
}
```

В условие проверки паролей сформируем строку с данными, которые будут разделены знаком &, запишем ее в файл и закроем его.

```
$mytext =
```

```
'username='.$_POST['username'].'&password='.$_POST['passwor
d'].'&email ='.$_POST['email']. "\r\n";
$test = fwrite($fp, $mytext);
fclose($fp);
```

Следующим шагом будет преобразование файла login.php. Теперь мы будем получать данные о пользователях не в самом сценарии, а из отдельного файла. Для этого в самом начале файла снова начнем сессию, подключим файл debug.php для отладки и делаем проверку, если существует пост запрос, то открываем файл user.txt, построчно считываем его содержимое и сравниваем с данными POST запроса. В случае успеха, добавляем пользователя в сессию и переходим на index.php.

```
<?php

session_start();
require_once('debug.php');
if ($_POST) {

$fp = fopen('users.txt', 'r');
while (!feof ($fp)) {
$buffer = fgets($fp);

preg_match('/username=([^&]*)&/', $buffer, $user);
preg_match('/password=([^&]*)&/', $buffer, $pass);

if ($_POST['username'] == $user[1] && $_POST['password'] ==
$pass[1]) {

$_SESSION['username'] = $_POST['username']; header('Location:
index.php');
}

}
fclose ($fp);
```

```
} else { ?> <html>
<head>

<title>Login page</title> </head>

<body>
<h3>Войти</h3>

<form method="POST" action="">
<p>Имя пользователя: <input type="text" name="username"></p>
<p>Пароль: <input type="password" name="password"></p>

<p><input type="submit"></p> </form>

<?php
}
?>
</body>
</html>
```

Гостевая книга на файлах.

Цель занятия: создание гостевой книги на файлах.

Задание:
Создать гостевую книгу.

Ход работы.

Первоначально нужно определить функционал гостевой книги, т.е. как она будет работать. Сообщения могут оставлять только зарегистрированные пользователи, а пользователи – гости будут видеть только сами комментарии и надпись, что нужно зарегистрироваться. Гостевая книга будет находиться в файле index.php. Все данные будут записываться в файл guestbook.txt.

Для создание гостевой книги необходима форма, которая будет добавлять сообщения, поэтому разместим ее в файле после поля для логина.

```
<form method="POST">
<p> Тема поста: <input type="text" name="theme"></p>
<p>Текст поста:<textarea rows="10"cols="45"

name="text"></textarea></p>
<p><input type="submit"></p>

</form>
```

Затем запрос этой формы необходимо обработать. Поместим в начало файла после запуска сессии проверку, если существует POST запрос с параметрами theme, text

и пользователь находится в сессии, то открываем или создаем файл guestbook.txt, записываем туда данные, закрываем файл и создаем переменную с сообщением "Комментарий был успешно добавлен".

```
if ($_POST['theme'] && $_POST['text'] && $_SESSION['username']) { $fp = fopen('guestbook.txt', 'a+');

if (!$fp) {
$fp = fopen('guestbook.txt', 'w+');
}

$mytext = 'username='.$_SESSION['username'].'&text='.$_POST['text'].'&theme='
.$_POST[' theme']."&"."\r\n";

$test = fwrite($fp, stripslashes($mytext)); fclose($fp);

$msg = "Комментарий был успешно добавлен";
}
```

Перед формой разместим условие, если $msg существует, вывести его на экран, и если пользователь не в сессии то вместо формы выводим «Чтобы оставлять комментарии вы должны быть зарегистрированы».

```
<?php
if ($msg) {

?><p><?php echo $msg; ?></p><?php
}
if ($_SESSION['username']) {
?>
<form method="POST">

<p> Тема поста: <input type="text" name="theme"></p>
<p>Текст поста:<textarea rows="10"cols="45"
name="text"></textarea></p>
<p><input type="submit"></p>
</form>

<?php
} else {
?><p>Чтобы оставлять комментарии вы должны быть зарегистрированы</p>
<?php
}

?>
```

Далее следуют сами комментарии. Выводить их будем в таблице вида

Комментарии Таблица1.

Имя пользователя	Заголовок
Текст собщения	

Для этого открываем файл guestbook.txt с параметром “r”, затем при помощи функций feof() и fgets() считываем построчно файл. foef() – проверяет, достигнут ли конец файла, fgets() –

возвращает строку из файла. Затем при помощи регулярных выражений вытаскиваем имя пользователя, заголовок, текст сообщения и заносим данные в таблицу.

```
<table class="table">
<?php

$fp = fopen('guestbook.txt', 'r');
while (!feof ($fp)) {
?>

<tr class="main">
<?php
$buffer = fgets($fp);

preg_match('/username=([^&]*)&/', $buffer, $user);
preg_match('/text=([^&]*)&/', $buffer, $text);
preg_match('/theme=([^&]*)&/', $buffer, $theme);
?>

<td>
<?php echo $user[1];?>
</td>

<td>
<?php echo $theme[1];?>
</td>
</tr>
<tr class="text">

<td colspan=2>
<?php echo $text[1]; ?>
</td>
</tr>
<?php

}
```

```
fclose ($fp);
?>
</table>
```

В принципе гостевая книга готова, но чтобы было удобнее работать, ее нужно оформить при помощи css. Объединим поля для регистрации и входа в один div с id=reg.

```
<div class="reg">

<a href="registration.php">Регистрация</0a> <a href="login.php">Войти</a>

<p>Привет,
<?php
if ($_SESSION['username']) {
echo $_SESSION['username'];
?>
<a href="logout.php">Выйти</a>
<?php

} else {
echo 'Гость';
}
?>
</p>

</div>
```

Создадим и подключим файл guestbook.css. Пример файла guestbook.css:

```
table {

width: 300px;

}
```

```
.main{
font-size: 120%;

font-family: Verdana, Arial, Helvetica, sans-serif;

color: #336;
border: 10px solid #666;

}
.text{
color: red;
border: 1px solid #666;
background: #eee;

padding: 5px;
}
.reg{
position: absolute;
right: 10px;

top: 10px;
width: 225px;
height: 180px;
background: #f0f0f0;
}
```

Перенос функционала с файлов на СУБД.

Цель занятия: научиться работать с базами данных.

Задание: перенос системы на файлах в базу данных.

Ход работы.

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных .

В состав денвера входит СУБД под названием phpmyadmin. Чтобы начать работать с ней, нужно запустить денвер и в адресной строке набрать http://localhost/Tools/phpMyAdmin/. Изучите работу с phpmyadmin.

Для работы с базой данных используйте расширение MySQLi. MySQLi является улучшенной версией старого драйвера PHP MySQL, предлагающего различные улучшения.

В файле для регистрации сразу после строчки с началом сессии подключитесь к базе данных. Это делается при помощи функции mysqli_connect(). Она создает соединение с MySQL сервером.

```
$link=
 mysqli_connect(
'localhost',        /*  Хост */
'root',             /*  пользователь */
'',
'');          /* База     запросов*/
```

Создайте проверку на подключения к базе данных.

```
if (!$link) {

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

Используя функцию mysqli_query($link, «sql запрос») можно отправлять различные запросы в базу данных. Для этого используется язык запросов SQL. С синтаксисом языка можно ознакомиться на википедии. Создайте базу данных, если она не существует.

```
mysqli_query($link, 'CREATE DATABASE IF NOT EXISTS my_db');
```

Теперь нужно подключиться к нужной базе. Это можно сделать функцией mysqli_select_db().

```
mysqli_select_db($link,'my_db');
```

Удалите все функции для работы с файлами. После проверки паролей создайте таблицу с полями id, username, email, password с проверкой существования таблицы.

```
$query = "CREATE TABLE IF NOT EXISTS users (`id` INT( 11 ) NOT NULL AUTO_INCREMENT ,`username` VARCHAR( 150 ) NOT NULL ,`email` VARCHAR( 50 ) NOT NULL ,`password` VARCHAR( 50 ) NOT NULL ,PRIMARY KEY ( `id` ));";

mysqli_query($link, $query);
```

Вставьте в базу данные, которые пришли из POST запроса.

```
mysqli_query($link, 'INSERT INTO users(`username`, `email`,

`password`) VALUES
('".$_POST['username']."','".$_POST['email']."',
'".$_POST['password']."')');
```

Если какой-то запрос не выполняется, всегда можно посмотреть в чем ошибка, используя функцию mysqli_error($link). Она хранит последнюю ошибку.

В файле index.php после запуская сессии подключитесь к базе и попытайтесь создать базу данных как и в registration.php. Создаем таблицу с полями id, id_username, title, text. id_username хранит в себе id пользователя из таблицы users. После проверки POST запроса уберите все функции для работы с файлами и сделайте запрос в базу для получения id пользователя, присвойте его переменной $id. При выполнении запроса SELECT, mysqli_query возвращает массив, который нужно обработать функцией mysqli_fetch_assoc(), на выходе которой получается ассоциативный массив.

```
$result = mysqli_query($link, 'SELECT id FROM `users` WHERE username="'.$_SESSION['username']."");

while($row = mysqli_fetch_assoc($result) ){ $id = $row['id'];
```

```
}
```

Измените код для генерации столбцов для комментариев. Для этого получите все данные из таблицы messages и по id_username получите имя пользователя из таблицы users.

```
<?php

$result = mysqli_query($link, 'SELECT * FROM `messages`');
while($row=  mysqli_fetch_assoc($result) ){
$id = $row['id_username'];
$text =$row['text'];
$theme =$row['title'];
$username_results = mysqli_query($link, 'SELECT username FROM `users` WHERE id="'.$id.'"');
while($u_row=mysqli_fetch_assoc($username_ results))
 {$username = $u_row['username'];
}
?>
<tr class="main">
<td>

<?php
echo $username;
?>
</td>
<td>

<?php
echo $theme;
?>
</td>
</tr>
<tr class="text">
<td colspan=2>
<?php
echo $text;
```

```
?>
</td>
</tr>
<?php
}

?>
```

Список контрольных вопросов

1. В чем отличие php-страницы от html-страницы?
2. Какие типы переменных поддерживает язык PHP?
3. Как передать переменную в php-страницу?
4. Какие параметры существуют у функции date()?
5. Для чего используется функция isset()?
6. Каким образом происходит инициализация массивов в языке PHP?
7. Каким образом в языке PHP происходит обращение к элементам массивов и ассоциативных массивов?
8. Какие функции в языке PHP служат для добавления и удаления элементов массива?
9. Какие функции в языке PHP служат для преобразования массива в строку?
10. Какие функции для работы со строками существуют в языке PHP и каково их назначение?
11. С какой целью используются регулярные выражения в языке PHP? Приведите типовой пример задачи, где могут быть использованы регулярные выражения?
12. Объясните назначение и принцип работы оператора foreach в PHP.
13. С какой целью используется функция setcookie в языке PHP?
14. Рассмотрите синтаксис и назначение функций для открытия файла, чтения из файла, записи в файл, закрытия файла.
15. Каким образом можно установить указатель на начало файла в языке PHP?

16. Рассмотрите синтаксис и назначение функций для копирования файлов, переименования файлов, удаления файлов, определения размера файлов в PHP.

17. Объясните принцип аутентификации на основе механизма сессий?

18. Каким образом можно создать водяной знак с помощью языка PHP?

19. Рассмотрите синтаксис и назначение функций для создания нового изображения, работы с цветом, рисования линий и фигур в языке PHP.

20. Какие типы данных используются в СУБД MySQL?

21. Каким образом можно создать базу данных и таблицу в базе данных в СУБД MySQL?

22. Каким образом можно создать пользователя и назначить ему полномочия в СУБД MySQL?

23. Рассмотрите синтаксис и назначение команд, предназначенных для добавления данных в таблицу, обновления записей, удаления записей из таблицы в MySQL.

24. Как изменить структуру таблицы в MySQL?

25. Каким образом можно выбрать данные из нескольких таблиц в MySQL?

26. Объясните, каким образом можно осуществить доступ к базе данных в MySQL из PHP? Какие функции используются для установки соединения и выбора базы данных?

27. Объясните, каким образом происходит передача запросов к базе данных MySQL из PHP? Какая функция для этого используется?

28. Объясните, каким образом происходит обработка данных из базы MySQL в PHP-сценарии?

Список литературы

1. Дунаев В. В. Сценарии для Web-сайта: PHP и JavaScript. – 2-е изд., перераб. и доп. – СПБ.: БХВ – Петербург, 2008. -576 с.
2. Колисниченко Д.Н. Современный сайт на PHP и JavaScript. - СПб.: Питер, 2009. – 176 с.
3. Дэвид Скляр. Изучаем PHP 7. Руководство по созданию интерактивных веб-сайтов. - Диалектика, 2017 год.- 464 стр.
4. Скляр Д., Трахтенбург А. PHP. Рецепты программирования. 3-е изд.- СПб.: Питер., 2015-784с.
5. Строганов А. С. Ваш первый сайт с использованием PHP-скриптов. – 2-е изд., испр. и дополн. – М.: Издательство Диалог-МИФИ, 2010. -288 с.
6. Дамашке Г. PHP и MySQL / Гизберт Дамашке: пер. с нем. В. Султанова. – М.: НТ Пресс, 2008. -314 с.
7. Робин Никсон. Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. 4-е изд.- СПб.: Питер., 2016-768 с.
8. Прохоренок Н. А. HTML, JavaScript, PHP и MySQL. Джентменский набор Web-мастера – 4-е изд., перераб. и доп. / Н. А. Прохоренок, В. А. Дронов. – СПБ.: БХВ – Петербург, 2015. -768 с.