Тема Сессии и куки

Работа с сессиями PHP

Что такое сессии в PHP?

Сессия - это механизм PHP, который позволяет хранить данные для конкретного пользователя между запусками скрипта.

Мы можем записывать какую-либо информацию в сессию и считывать ее оттуда в следующем запуске этого или другого скрипта сайта.

С помощью сессии можно реализовать авторизацию пользователей, корзину интернет-магазина и другое.

Инициализируем сессию

Чтобы записать что-то в сессию ее сначала нужно *инициализировать* с помощью функции **session_start()**:

```php
<?php
	//Инициализируем сессию:
	session_start();
?>
```

Обратите внимание на то, что в одном скрипте сессия должна инициализироваться **только один раз**, иначе скрипт выдаст ошибку.

Пишем в сессии

После инициализации мы можем записать что-нибудь в сессию.

Сделать это не сложно:

```php
<?php
	//Инициализируем сессию:
	session_start();

	//Пишем в сессию:
	$_SESSION['test'] = 'Тест!';
?>
```

Читаем из сессий

После того, как мы что-то записали в сессию, мы можем это оттуда извлечь:

```php
<?php
	//Инициализируем сессию:
	session_start();

	//Выведем переменную test из сессии:
	echo $_SESSION['test'];
?>
```

Возможные проблемы

Основная проблема при работе с сессией следующая: нельзя делать никакого вывода в браузер до окончания работы с сессиями.

Что такое вывод в браузер?

Это любой символ до **<?php**, например, текст или тег, даже пробел. Кроме того нельзя делать выводы через **echo**, **var_dump** и **print_r**.

Кодировка вашего документа обязательно должна быть utf-8 **без BOM**. Если она будет просто utf-8, то перед <?php будет вставлен спец. символ, характерный для данной кодировки и сессии работать не будут.

Если вы делаете какие-то вывод в браузер до работы с сессией, то увидите следующую ошибку: *Warning: Cannot send session cookie - headers already sent*.

Эта ошибка также вызывается другими сообщениями об ошибках, так как эти сообщения - это вывод в браузер до старта сессии. Обратите на это внимание.

Сделаем счетчик на сессиях

Чтобы продемонстрировать работу с сессиями давайте реализуем **счетчик** количества обновления страницы пользователем.

При каждом обновлении страницы он будет увеличиваться на единицу, а при закрытии браузера - обнуляться (после закрытия нужно подождать 15-25 минут):

```php
<?php
	//Инициализируем сессию:
	session_start();
	/*
		Переменная $_SESSION['counter'] будет нашим счетчиком.
```

```
        Если скрипт запускается первый раз -
        она будет пуста, присвоим ей единицу.
        Если не первый раз - тогда прибавим единицу.
    */
    if (!isset($_SESSION['counter'])) {
        $_SESSION['counter'] = 1;
    } else {
        $_SESSION['counter'] = $_SESSION['counter'] + 1;
    }

    //Выведем значение счетчика:
    echo 'Вы обновили эту страницу '.$_SESSION['counter'].' раз!';

    /*
        Обновите страницу несколько раз,
        посмотрите на то, как увеличивается значение переменной.
        Затем закройте браузер, подождите полчаса и откройте сно
ва -
        убедитесь в том, что переменная обнулилась!
    */
?>
```

Удалении переменных из сессии

Переменную сессии можно удалять с помощью функции **unset**:

```
<?php
    //После выполнения этой команды в $_SESSION['var'] станет null
:
    unset($_SESSION['var']);
?>
```

Завершение сессии

Если же вам нужно удалить все переменные сессии для данного пользователя, то вместо **unset** следует воспользоваться функцией **session destroy()** (ее можно вызывать только тогда, когда сессия запущена через session_start):

```
<?php
	//После выполнения этой команды ВСЕ переменные сессии станут n
ull:
	session_destroy();
?>
```

Мы рассмотрели принудительное завершение сессии. Однако сессия может завершится сама в двух следующих случаях: при закрытии браузера пользователем или по истечении определенного времени, во время которого пользователь не совершает никаких действий на странице (по умолчанию это время около 15-25 минут).

Какие задачи решает сессия

Стандартные задачи, которые решают с помощью сессии, это корзина интернет-магазина, форма для заполнения пользователем, разбитая на несколько страниц сайта (на первой мы спрашиваем имя, на второй email и т.д.)

Сессии и формы

Поможет с решением задач: https://youtu.be/GDFYbddLH5A

Самостоятельная работа

Работа с сессиями

1. По заходу на страницу запишите в сессию текст 'test'. Затем обновите страницу и выведите содержимое сессии на экран.
2. Пусть у вас есть две страницы сайта. Запишите на первой странице что-нибудь в сессию, а затем выведите это при заходе на другую страницу.
3. Сделайте **счетчик обновления страницы** пользователем. Данные храните в сессии. Скрипт должен выводить на экран количество обновлений. При первом заходе на страницу он должен вывести сообщение о том, что вы еще не обновляли страницу.
4. Сделайте две страницы: **index.php** и **test.php**. При заходе на **index.php** спросите с помощью формы страну пользователя, запишите ее в сессию (форма при этом должна отправится на эту же страницу). Пусть **затем** пользователь зайдет на страницу **test.php** - выведите там страну пользователя.
5. Запишите в сессию время захода пользователя на сайт. При обновлении страницы выводите **сколько секунд назад** пользователь зашел на сайт.
6. Спросите у пользователя **email** с помощью формы. Затем сделайте так, чтобы **в другой форме** (поля: имя, фамилия, пароль, email) при ее открытии поле email было автоматически заполнено.

Работа с cookie на PHP

Cookie (куки) - это способ долговременного хранения данных в браузере пользователя.

К сожалению, в куки можно сохранить только 4 килобайта информации. Кроме того, есть ограничение на количество кук для данного домена.

Пишем в cookie

Обращаю ваше внимание на то, что в куки нужно писать до любого вывода на экран. Замечания по этому поводу аналогичны проблемам с сессиями на PHP.

Написать что-то в куки можно с помощью функции **setcookie**, которая первым параметром принимает имя этой куки, а вторым - значение:

```
<?php
      //Запишем в куки с именем test значение 'Тест!':
      setcookie('test', 'Тест!');
 ?>
```

Однако такие куки долго не живут - только до закрытия браузера.

Продлить время жизни куки можно с помощью третьего параметра, который принимает **время** (конкретную дату) окончания жизни куки в формате timestamp.

Для тех, кто не помнит, что это за формат - напоминаю: это количество секунд, прошедших с первого января 1970 года.

Однако устанавливать конкретную дату 'смерти' куки не очень удобно, так как дата установки этой куки всегда разная.

Поэтому третий параметр принято записывать так: *настоящий момент времени + N секунд*.

Настоящий момент времени в формате **timestamp** можно получить с помощью функции time. Примеры:

```
<?php
      //Запишем куку на час (в часе 3600 секунд!):
      setcookie("test","Тест!", time() + 3600);
```

```php
    //Запишем куку на день (в сутках 3600*24 секунд!):
    setcookie("test","Тест!", time() + 3600*24);

    //Запишем куку на месяц (в месяце 3600*24*30 секунд!):
    setcookie("test","Тест!", time() + 3600*24*30);

    //Запишем куку на год (в году 3600*24*30*365 секунд!):
    setcookie("test","Тест!", time() + 3600*24*30*365);
?>
```

Читаем из cookie

Куки можно прочитать с помощью глобального массива **$_COOKIE**.

Давайте прочитаем ранее установленную куку **test**:

```php
<?php
    //Выведем на экран значение куки test:
    echo $_COOKIE['test'];
?>
```

Удаляем cookie

Удаляют куки очень хитрым способом - устанавливая дату 'смерти' куки на текущий момент времени:

```php
<?php
    //Удалим куку, установив третий параметр в текущий момент врем
ени:
    setcookie('test', '', time());
?>
```

Задачи на Cookie

Задание 1: Инициализируйте переменную для подсчета количества посещений. Если соответствующие данные передавались через cookie сохраняйте их в эту переменную. Нарастите счетчик посещений. Инициализируйте переменную для хранения значения последнего посещения страницы. Если соответствующие данные передавались из cookie , отфильтруйте их и сохраните в эту переменную. Установите соответствующие cookie

Задание 2: Выводите информацию о количестве посещений и дате последнего посещения

```
<?php
//инициализируем переменную для подрасчета
//количества посещений
// Если соответсвующие данные передавались через cookie
//сохраняем их в эту переменную
$visit_counter = 0;
if (isset($_COOKIE['visitCounter']) &&
     is_numeric($_COOKIE['visitCounter'])){
    $visit_counter = $_COOKIE['visitCounter']*1;
                                        }
// Прирациваем счетчик посещений
$visit_counter++;
//Инициализруем переменную для хранения значения последнего посещения
// Если соответствующие данные передавались из cookie, сохраняем их в эту переменную
$last_visit = '';
if (isset($_COOKIE['lastVisit'])){
    $last_visit=htmlspecialchars($_COOKIE['lastVisit'],
    ENT_QUOTES);
$last_visit=stripslashes(trim($last_visit));
}
//устанавливаем cookie
setcookie ('visitCounter', $visit_counter, 0x7FFFFFFF) ;
setcookie ('lastVisit', date ('d/m/Y H:i:s'), 0x7FFFFFFF);
//Выводим информацию о количестве посещений и дате последнего
посещения
if($visit_counter == 1) {
    print '<h2>Добро пожаловать!</h2>';
} else {
print <<<HTML
<h2> Вы здесь уже $visit_counter раз</h2>
<p>Последнее сообщение: $last_visit</p>
HTML;
}
?>
```

9) Задачи на Session в PHP

Задача. Сделайте две страницы: **index.php** и **hello.php**.

При заходе на **index.php** спросите с помощью формы имя пользователя, запишите его в **куки**.При заходе на **hello.php** поприветствуйте пользователя фразой **"Привет, %Имя%!"** .

Решение:

Страница index.php:

```
<form action="" method="GET">
<input type="text" name="username">
<input type="submit">
```

```
</form>
<?php
7. //Если форма была отправлена и имя не пустое:
if (!empty($_REQUEST['username'])) {
//Пишем имя в куки:
setcookie('username', $_REQUEST['username'], time()+3600, '/');}
?>
```

Страница hello.php:

```
<?php
//Если есть данные в куки об имени пользователя:
if (!empty($_COOKIE['username'])) {
    echo $_COOKIE['username']; //выведем имя на экран
}
?>
```

Спросите у пользователя телефон с помощью формы. Затем сделайте так, чтобы **в другой форме** (поля: имя, фамилия, телефон) при ее открытии поле телефон было автоматически заполнено.

Спрашиваем телефон:

```
<form action="" method="GET">
<input type="text" name="phone">
<input type="submit">
</form>
<?php
//Если форма была отправлена и телефон не пустой:
if (!empty($_REQUEST['phone'])) {
//Пишем телефон в куки:
setcookie('phone', $_REQUEST['phone'], time()+3600, '/');
}
?>
```

Другая форма:

```
<?php
//Если телефон есть в куки – запишем его в переменную $phone:
if(!empty($_COOKIE['phone']))
$phone = $_COOKIE['phone'];
else
$phone = '';
?>
<form action="" method="GET">
<input type="text" name="name">
<input type="text" name="surname">
<!– Заполним атрибут value переменной $phone: –>
<input type="text" name="phone" value="" <?php echo $phone ?> "> 15.

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

Задание 1 .

Создайте три страницы с формами, содержащими по одному полю ввода на стр. Обработчик каждой последующей формы будет вести на следующую страницу. На первой страницы спросим имя, на второй странице фамилию, на третьей возраст, а на четвертой выведите все данные используя сессию.

Задание 2.

Создайте в сессии массив для хранения всех восещенных страниц и сохраните в качестве его очередного элемена путь к текущей странице. Выведите в цикле список всех посещенных пользователями страниц

Page 1

```
<?php
// page 1
session_start(); //открываем сессию
//создаем в сессии массив для хранения
// всех посещенных страниц
//и сохраняем в качестве его очередного
//элемента текущую страницу
$_SESSION['pages'][]=$_SERVER['PHP_SELF'];
?>
<form action="page2.php" method="post">
Ваше Имя:<br><br>

<input type="text" name="name"><br><br>
<input type="submit" value="дальше">
</form>
Page 2
<?php
//страница 2
session_start(); //открываем сессию
$_SESSION['pages'][] = $_SERVER['PHP_SELF'];
//если пользователь ввел имя, фильтруем его и заносим в сессию
if(isset($_POST[name])) {
$_SESSION[name]=strip_tags(trim($_POST[name]));
}
?>
<form action="page3.php" method="post">
Фамилия: <br> <br>
<input type="text" name="Fname"> <br><br>
<input type="submit" value="дальше">
</form>
Page 3 & 4
<?php
//page3.php
session_start(); //открываем сессию
$_SESSION['pages'][] = $_SERVER['PHP_SELF'];
//если пользователь ввел имя, фильтруем его и заносим в сессмю
if(isset($_POST[Fname])){
$_SESSION[Fname]=strip_tags(trim($_POST[Fname]));
?>
<form action="page4.php" method="post">
Возраст:<br><br>
<input type="text" name="age"> <br><br>
<input type="submit" value="дальше">
</form>
<?php
// page 4
session_start(); //открываем сессию
$_SESSION ['pages'] [] = $_SERVER['PHP_SELF'];
//если пользователь ввел имя, фильтруем его и заносим в сессию
if(isset($_SESSION[name]) && isset ($_SESSION[Fname])
$$ isset ($_SESSION[age]))
{ echo "Your name $_SESSION[name] $_SESSION[Fname] <br>"; 25.
echo "Ваш возраст $_SESSION[age] лет";
}
```

```
if(isset($_SESSION['pages']))
{print '<h3> Список посещенных страниц</h3>';
//вывод всех посещенных страниц в цикле
print '<ol>';
foreach ($_SESSION['pages'] as $page)
{
echo '<li>', $page, '</li>';
}
print '</ol>';
}
?>
```

Самостоятельная работа

Работа со cookie

7. По заходу на страницу запишите в куку с именем **test** текст '123'. Затем обновите страницу и выведите содержимое этой куки на экран.
8. Удалите куку с именем **test**.
9. Сделайте **счетчик посещения сайта** посетителем. Каждый раз, заходя на сайт, он должен видеть надпись: **'Вы посетили наш сайт % раз!'**.
10. Спросите **дату рождения** пользователя. При следующем заходе на сайт напишите сколько дней осталось до его дня рождения. Если сегодня день рождения пользователя - поздравьте его!