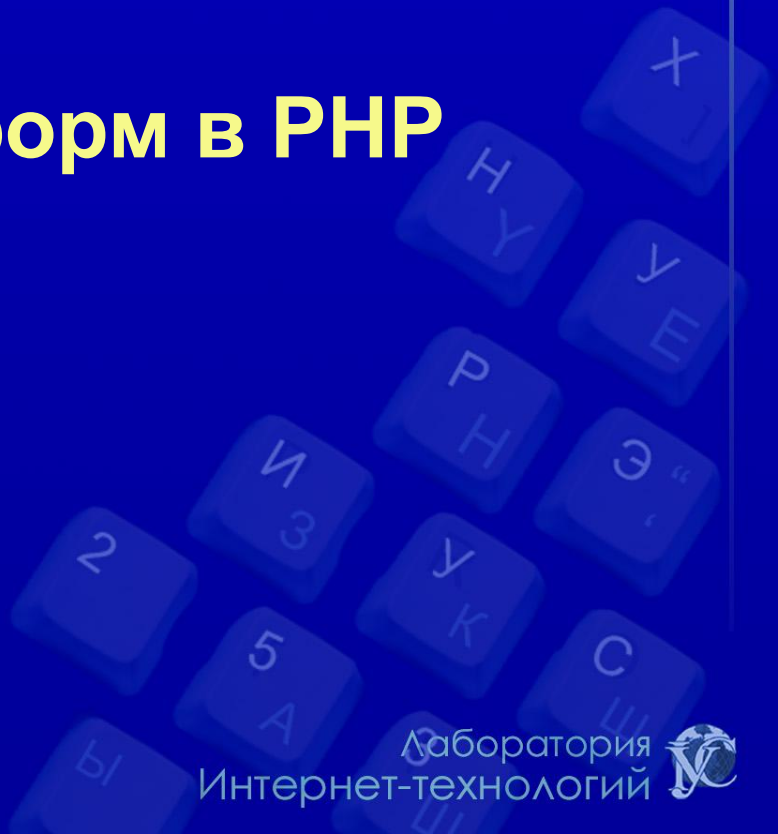
Лекция №3
на тему:

Функции.
Обработка HTML-форм в PHP



Объявление и вызов функции

```
function fun ()
{
// операторы
}
```

- Если функция возвращает значение, в ее теле должен присутствовать оператор return.

```
function fun ()
{
// операторы
return $var;
}
```

Пример функции, возвращающей сумму трех параметров

```
function sum ($a, $b, $c)
{
$var = $a + $b + $c ;
return $var;
}

echo sum (1,2,3);
```

Условное объявление функции

```
$flag = true;
// если переменная flag = true, то объявляем функцию
if ($flag)
{
    function sum ($a, $b, $c)
    {
    $var = $a + $b + $c ;
    echo $var;
    }
}
// вызываем функцию, если переменная flag = true
if ($flag) sum (1,2,3);
```

Передача параметров по значению и ссылке

Пример передачи аргумента по значению.

```
function sum ($var)
{
$var = $var + 2 ;
return $var;
}
    $per = 10;
    echo sum($per);
    echo “$per”;
```

Пример передачи аргумента по ссылке

```
function sum (&$var)
{
   $var = $var + 2 ;
   return $var;
}
   $per = 10;
   echo sum($per);
   echo “$per”;
```

Необязательные параметры

```
function sum ($a=5, $b=7)
{
$var = $a + $b;
echo $var;
}
    sum();
    sum(3);
    sum(1,2);
```

- Некорректное объявление функции:

```
function sum ($a=5, $b)
```

- Если после этого вызывать функцию sum() :

```
sum(1);
```

то прекращения выполнения скрипта не происходит

Переменное количество параметров

- Функции без параметров можно передавать любое их количество.

```
function sum (){
echo “Вызов функции”;
}
   sum(1, 17, “Третий параметр”);
```

- При таком вызове интерпретатор PHP не выдает предупреждений

Функции, работающие с переменным количество параметров

- func_num_args() возвращает количество параметров, переданных функции
- func_get_args() возвращает массив с параметрами, переданными функции
- func_get_arg($arg_num) возвращает значение параметра с номером $arg_num.

- Пример функции, которая выводит свои параметры.

```
function f() {
for ($i=0; $i < func_num_args(); $i++)
echo “Параметр номер”. $i . func_get_arg($i) . “<br>”   ;
// вызов функции
echo f (1, 17, “Третий параметр”);
```

Локальные переменные

- Переменные в функциях имеют локальную область видимости.
- Пример.

```
function f()
 {
   $var=7;
   echo $var;
}
   $var=15; // внешняя переменная
   f();
   echo $var;
```

Глобальные переменные

- Если переменная объявлена как global, доступ к ней возможен из любой части программы.

- Пример.

```
function f()
 {
    global $var;
    $var=7;
    echo $var;
}
$var=15;
echo $var;
f();
```

Статические переменные

- Локальная переменная при каждом вызове функции инициализируется заранее.
- Чтобы локальная переменная сохраняла свое предыдущее значение при новых вызовах функции, ее можно объявить статической при помощи ключевого слова static.

- Пример.

```
function f()
 {
    static $var=0;
    return ++$var;
}
```

Обработка HTML-форм. Протокол GET

- Если используется метод GET, то передача параметров происходит в строке запроса после символа вопроса.

http://www.mysite.ru/forum/read.php?id_forum=1

- GET-параметры автоматически помещаются в суперглобальный массив $_GET. Имена параметров выступают в качестве ключей массива.
-
- Пример. Выведем значение GET-параметра id_forum.
- <?php
- echo $_GET[‘id_forum’];
- ?>

GET-параметры

- HTTP допускает использование в качестве имени GET-параметра имена, начинающиеся с цифры или содержащие тире.
- Значения таких параметров нужно извлекать из переменной окружения $_SERVER['QUERY_STRING']

- Если скрипту нужно передать несколько GET-параметров:

http://www.mysite.ru/forum/read.php?id_forum=1&id_theme=3&id_post=7

- Все параметры находятся в массиве $_GET.

Функции преобразования GET-параметров

- GET-параметры и их значения могут содержать символы кириллицы. Эти символы нужно преобразовать в другой формат. Для этого в PHP существует несколько функций.

- urlencode($str) – возвращает строку, в которой недопустимые символы кодируются знаком % и двумя 16-ричными числами. Пробел кодируется знаком +.

- urldecode($str) – обратная urlencode() функция, выполняющая декодирование символов, начинающихся с %.

- Для разбора строки запроса предназначена функция parse_url(), которая возвращает отдельные компоненты запроса в виде ассоциативного массива.

Пример

```
<?php
$url = ‘http://user:pass@www.mysite.ru/path/index.php?par=value#serg’;
$arr = parse_url($url);
    echo “<pre>”;
    print_r($arr);
    echo “</pre>”;
?>
```

- Результат:

```
Array (
[scheme] => http
[host] => www.mysite.ru
[user] => user
[pass] => pass
[path] => /path/index.php
[query] => par=value
[fragment] => serg
)
```

Пример программы обработки ввода пользователя в HTML-форму

- Создадим файл form.html:

```
<html>
<body>
<form action="form.php" method="GET">
<input type="text" name="user”>
<br>
<textarea name="address" rows="5" cols="40">
</textarea>
<br>
<input type="submit" value=”hit it!">
</form>
</body>
</html>
```

Программа, обрабатывающая данные формы

- Создадим файл form.php:

```
<html>
<head>
<title> Чтение данных формы  </title>
</head>
<body>
    <?php
     print "Welcome <b>$user</b><P>\n\n";
     print "Your address is:<P>\n,\n<b>$address</b>";
     ?>
</body>
</html>
```

Обработка формы с флажками

```
<form method=’POST’ action=’flag.php’>
<input type=’checkbox’ name=’php’ checked> Вы знакомы с PHP?
<input type=’checkbox’ name=’c’ checked> Вы знакомы с C++?
<input type=’checkbox’ name=’pasc’ > Вы знакомы с Паскалем?
<input type=’checkbox’ name=’basic’> Вы знакомы с VBA?
< input type=’submit’ value=’Передать’>
</form>
    <?php
    echo “<pre>”;
    print_r($_POST);
    echo “</pre>”;
    ?>
```

- Результат:

```
Array (
 [php] => on
[c] => on
)
```

Доступ ко всем полям формы через ассоциированный массив

В зависимости от того, какой метод передачи используется - GET или POST- используется один из массивов — $_GET или $_POST. Это ассоциированные массивы, содержащие пары имя/значение для каждого элемента переданной формы.

Пример. Вывести список всех полей и переданных значений при использовании метода передачи GET.

```
<html> <head> <title> Чтение данных произвольной формы с помощью ассоциированных массивов </title> </head>
<body>
<?php
    foreach ( $_GET as $key=>$value ) {
    print "$key = = $value<BR>\n";
    }
?>
</body>
 </html>
```

Определение метода передачи

Пример:

```
<?php
$PARAMS = ( isset( $_POST ) )
? $_POST : $_GET;
foreach ( $PARAMS as $key=>$value )
{
if ( gettype( $value ) == "array" )
{
print "$key == <br>\n";
foreach ( $value as $two_dim_value)
print "$two_dim_value<br>";
}
else
print "$key == $value<br>\n";
  } }
?>
```