Алгоритмирование и программирование

Лекция 4

Функции PHP

Мы уже в наших занятий применяли многие встроенные функции, входящие в набор функций PHP, например fopen() и assort(), substr() и многие другие.

Мы познакомились с функциями подключения к базе данных mysql_connect(), функция отправки запроса mysql_query (), и так далее. Этих функций разного назначения в PHP очень много, они используются для решения различных задач. Их описание и примеры использования можно посмотреть на официальном сайте PHP по ссылке https://www.php.net/manual/ru/funcref.php

Это встроенные функции, но язык PHP предоставляет нам возможность создавать собственные функции.

Сейчас важно понять, что существуют два отдельных аспекта:

Создание функции PHP

Вызов функции PHP

Создание функции PHP

Собственную PHP-функцию создать очень просто. Предположим, вы хотите создать функцию PHP, которая просто выводит в браузере короткое сообщение, когда вы ее вызываете. В следующем примере мы создаем функцию writeMessage(), а затем вызывает ее сразу после создания.

Обратите внимание, что при создании функции ее имя должно начинаться с ключевого слова function, и весь код PHP должен быть помещен внутри скобок { }, как в приведенном ниже примере:

```
<html>

<head>
  <title>Writing PHP Function</title>
</head>

<body>

  <?php
    /* Определение функции  PHP  */
    function writeMessage() {
      echo "<h1>Вы прекрасны! Хорошего дня!</h1>";
    }

    /*Вызов функции PHP */
    writeMessage();
  ?>

</body>
```

```
</html>
```

Этот код выводит следующий результат — **Вы прекрасны! Хорошего дня!**

Функции PHP с параметрами

PHP дает вам возможность передавать собственные параметры внутри функции. Вы можете передать столько параметров, сколько вам нужно. Эти параметры работают как переменные внутри функции. В следующем примере мы берем два целочисленных параметра и суммируем их, а затем выводим.

```
<html>

 <head>
  <title>Writing PHP Function with Parameters</title>
 </head>

 <body>

  <?php
    function addFunction($num1, $num2) {
      $sum = $num1 + $num2;
      echo "Сумма двух чисел равна: $sum";
    }

    addFunction(10, 20);
  ?>

 </body>
</html>
```

Этот код выводит следующий результат — **Сумма двух чисел равна: 30**

Передача аргументов по ссылке

В функции можно передавать аргументы по ссылке. Это означает, что ссылка на переменную управляется функцией, а не экземпляром значения переменной.

Любые изменения, внесенные в аргумент в этих случаях, изменяют значения исходной переменной. Вы можете передать аргумент по ссылке, добавив к имени переменной амперсанд (&) либо в вызове функции, либо в определении функции. В следующем примере продемонстрированы оба случая.

```
<html>

 <head>
  <title>Passing Argument by Reference</title>
 </head>

 <body>

  <?php
    function addFive($num) {
      $num += 5;
    }

```

```
   function addSix(&$num) {
     $num += 6;
   }

   $orignum = 10;
   addFive( $orignum );

   echo "Исходное значение равно $orignum<br />";

   addSix( $orignum );
   echo "Исходное значение равно $orignum<br />";
  ?>

 </body>
</html>
```

Этот код отображает следующий результат —

```
Исходное значение равно 10
Исходное значение равно 16
```

Функции PHP, возвращающие значение

Функция может возвращать значение с помощью оператора return в сочетании со значением или объектом. Return останавливает выполнение функции и отправляет значение обратно вызывающему коду. Более одного значения вы можете вернуть из функции с использованием массива return (1,2,3,4).

В следующем примере мы принимаем два целочисленных параметра и суммируем их, а затем возвращаем сумму в вызывающую программу. Обратите внимание, что ключевое слово return используется для возврата значения из функции.

```
<html>

 <head>
  <title>Writing PHP Function which returns value</title>
 </head>

 <body>

  <?php
   function addFunction($num1, $num2) {
     $sum = $num1 + $num2;
     return $sum;
   }
   $return_value = addFunction(10, 20);

   echo "Результат, возвращаемый функцией: $return_value";
  ?>

 </body>
</html>
```

Этот код выводит следующий результат: **Результат, возвращаемый функцией: 30**

Установка для параметров функций значений по умолчанию

Вы можете установить для параметра значение по умолчанию, если вызывающий функцию объект не передает его. Следующая функция выводит NULL в случае, если в эту функцию не было передано ни одного значения.

```
<html>

 <head>
  <title>Writing PHP Function which returns value</title>
 </head>

 <body>

  <?php
    function printMe($param = “Текст, заданный по умолчанию”) {
      print $param;
    }

    printMe("Это тест");
    printMe();
    echo “<br>”;
  ?>

 </body>
</html>
```

Это код выводит следующий результат — **Это тест**
Текст, заданный по умолчанию

Динамические вызовы функций

Имена функций можно назначать, как строки в переменных, а затем обрабатывать эти переменные точно так же, как и имя функции. В следующем пример продемонстрировано такое поведение.

```
<html>

 <head>
  <title>Dynamic Function Calls</title>
 </head>

 <body>

  <?php
    function sayHello() {
      echo "Привет!<br />";
    }

    $function_holder = "sayHello";
    $function_holder();
  ?>

 </body>
</html>
```

Этот код выводит следующий результат: Привет!

Функции, определяемые пользователем

Для чего нужны функции? Чтобы ответить на этот вопрос, нужно понять, что вообще представляют собой функции. В программировании, как и в математике, функция есть отображение множества ее аргументов на множество ее значений. То есть функция для каждого набора значений аргумента возвращает какие-то значения, являющиеся результатом ее работы. Зачем нужны функции, попытаемся объяснить на примере. Классический пример функции в программировании – это функция, вычисляющая значение факториала числа. То есть мы задаем ей число, а она возвращает нам его факториал. При этом не нужно для каждого числа, факториал которого мы хотим получить, повторять один и тот же код – достаточно просто вызвать функцию с аргументом, равным этому числу.

Функция вычисления факториала натурального числа

```
<?php
function fact($n)
{
  if ($n==0) return 1;
  else return $fact = $n * fact($n-1);
}
echo fact(3).'<br>';
   // можно было бы написать echo (3*2);
   // но если число большое,
echo fact(50).'<br>';
   // то удобнее пользоваться функцией,
   // чем писать echo (50*49*48*...*3*2);
   echo fact(5).'<br>';
?>
```

Обратите внимание, здесь функция рекурсивно вызывает сама себя, до тех пор, пока не выполнит все умножения.

Даже цикл не требуется.

Посмотрим, как в общем виде выглядит задание (объявление) функции. Функция может быть определена с помощью следующего синтаксиса:

Функция

```
function Имя_функции (параметр1, параметр2, … параметрN)
{
  Блок_действий
  return "значение возвращаемое функцией";
}
```


Когда аргумент передается в функцию по значению, изменение значения аргумента внутри функции не влияет на его значение вне функции. Чтобы позволить функции изменять ее аргументы, их нужно передавать по ссылке. Для этого в определении функции перед именем аргумента следует написать знак амперсанд «&».

Рассмотрим ещё один пример передачи аргумента по ссылке и напишем функцию добавления элементу формы типа радио параметра checked/

```php
<?php
// напишем функцию, которая бы добавляла
// к строке слово checked
function add_label(&$data_str){
  $data_str .= "checked";
}
$str = "<input type=radio name=article ";
  // пусть имеется такая строка
echo $str .="><br>";
        // выведет элемент формы –
        // не отмеченную радио кнопку
    add_label($str);
        // вызовем функцию
    echo $str .="><br>";
        // это выведет уже отмеченную
        // радио кнопку
    ?>
```