Работа с функциями в PHP

Пользовательские функции, рекурсия, анонимные функции

LoftSchool
от мыслителя к создателю

Понятие функции

- Функция – программный блок, который может многократно выполняться в любом месте сценария

```
// описание функции
function myFunction(){
    echo "<h1>Моя функция!</h1>";
}
```

```
// вызов функции
myFunction();
```

Правила использования

- Внутри функции можно использовать любой корректный PHP-код, в том числе другие функции
- Корректное имя функции начинается с буквы или знака подчеркивания, за которым следует любое количество букв, цифр или знаков подчеркивания
- PHP не поддерживает перегрузку функции, также отсутствует возможность переопределить или удалить объявленную ранее функцию
- Функции не обязаны быть определены до их использования, исключая тот случай, когда функции определяются условно. В этом случае, обработка описания функции должна предшествовать ее вызову

Проверка существования функции

- Для того, чтобы PHP не вывел ошибку о том, что функция с таим именем уже существует, можно использовать функцию **function_exists**

```
if ( !function_exists('someFunction') ) {
    function someFunction(){
        echo "My function";
    }
}
```

Функции зависящие от условий

```
superFunction();
$marker = true;
if($marker){
    function superTwoFunction(){
        echo "Функция, зависящая от условий<br />";
    }
}
superTwoFunction();
function superFunction(){
    echo "Эта функция всегда видна с самого начала программы<br
/>";
}
```

LoftSchool
от мыслителя к создателю

Функции зависящие от условий

```
outFunction();
inFunction();

function outFunction(){
    function inFunction(){
        echo "Внутренняя функция";
    }
}
```

LoftSchool
от мыслителя к создателю

Область видимости переменных

```php
function myFunctionSayHey($name){
    echo "<h1>Привет $name!</h1>";
    $name = "Семен";
}
    myFunctionSayHey("Владимир"); // ??
    $name = "Mike";
    myFunctionSayHey($name); // ??
    echo $name; // ???
```

LoftSchool
от мыслителя к создателю

Аргументы функций

```php
function sayHeyMessage($name){
    echo "<h1>Привет $name!</h1>";
}
    sayHeyMessage("Василий"); // ??
    $user = "Михаил"; // ??
    sayHeyMessage($user); // ??
    $func = "sayHeyMessage"; // ??
    $func("Гость"); // ??
```

LoftSchool
от мыслителя к создателю

Аргументы функций по умолчанию

```
function sayHeyMessage($name = “Гость”){
    echo "<h1>Привет $name!</h1>";
}
    sayHeyMessage("Василий"); // ??
    sayHeyMessage(); // ??
```

Обращение к глобальной области видимости #1

```php
function superFunction($name){
    echo "<h1>Привет $name!</h1>";
    global $name;
    $name = "Вася";
}

$name = "Mike";
superFunction($name);
echo $name; // Вася
```

LoftSchool
от мыслителя к создателю

Обращение к глобальной области видимости #2

```php
function superFunction($name){
    echo "<h1>Привет $name!</h1>";
    $GLOBALS['name'] = "Вася";
}

$name = "Mike";
superFunction($name);
echo $name; // Вася
```

LoftSchool
от мыслителя к создателю

Передача аргументов по ссылке #1

```
function superFunction($name) {
        echo "<h1>Привет $name!</h1>";
        $name = "Вася";
}
        sayHello("John");
        $name = "Mike";
        superFunction($name);
        superFunction(&$name);  // ошибка в PHP с версии 5.3
```

Передача аргументов по ссылке #2

```php
function superFunction(&$name) {
    echo "<h1>Привет $name!</h1>";
    $name = "Вася";
}
    superFunction("John"); // ошибка в PHP с версии 5.3
    $name = "Mike";
    superFunction($name); // ?
    superFunction($name); // ?
    superFunction(&$name); // ошибка в PHP с версии 5.3
```

LoftSchool
от мыслителя к создателю

Статические переменные

```php
// обычная функция
function simpleFunc(){
    $a = 0;
    echo $a++;
}
simpleFunc(); // 0
simpleFunc(); // 0
simpleFunc(); // 0
```

```php
// функция со статическими переменными
function simpleStaticFunc(){
    static $a = 0;
    echo $a++;
}
simpleStaticFunc(); // 0
simpleStaticFunc(); // 1
simpleStaticFunc(); // 2
```

LoftSchool
от мыслителя к создателю

Возвращение значений

```
function square($num) {
    return $num * $num;
    // Этот код никогда не исполнится
    echo "Мертвый код";
}
    echo square(4); // выводит 16
    $result = square(4);
```

LoftSchool
от мыслителя к создателю

Возвращение массивов

```
function numbers() {
    return array (0, 1, 3, 2);
}

list($zero, $one, $two) = numbers();
```

Возвращение массивов

```
function numbers() {
    return array (0, 1, 3, 2);
}

echo numbers()[1];
```

LoftSchool
от мыслителя к создателю

Рекурсивные функции

```
function recursive($a){
    if($a < 20){
        echo $a.'<br />';
        recursive($a + 1);
    }
}
```

LoftSchool
от мыслителя к создателю

Использование аргументов переменной длины

- func_num_args() - возвращает количество аргументов в функции
- func_get_arg(n) - возвращает значение n-го аргумента функции
- func_get_args() - возвращает массив аргументов функции

LoftSchool
от мыслителя к создателю

Анонимные функции

```php
$greet = function($name){
    echo "Привет, {$name}!<br />";
};

$greet('World');
$greet('PHP');
```

Замыкания

```
function setDollar($dollar){
    return function($dollars) use ($dollar){
        return $dollars * $dollar;
    };
}

$countRubles = setDollar(61);
echo $countRubles(15);
```

LoftSchool
от мыслителя к создателю

Инструкции включения скриптов

- **include** - в случае ошибки генерирует предупреждение E_WARNING и продолжает исполнение кода
- **require** - в случае ошибки генерирует ошибку E_COMPILE_ERROR и останавливает исполнение кода
- **include_once** - в случае ошибки, действия аналогичны include
- **required_once** - в случае ошибки, действия аналогичны require

Включение скриптов с возвратом результата

return.php

```php
<?php
    $var = 'PHP';
    return $var;
?>
```

noreturn.php

```php
<?php
    $var = 'PHP';
?>
```

testreturns.php

```php
<?php
    $foo = include 'return.php';
    echo $foo; // Выводит 'PHP'
    $bar = include 'noreturn.php';
    echo $bar; // Выводит 1
?>
```

LoftSchool
от мыслителя к создателю

Что нужно сделать после вебинара?

1. Пересмотреть запись вебинара
2. Прочитать методичку
3. Продолжить чтение книг из списка “рекомендованной литературы”
4. Выполнить домашнее задание
5. Задавать свои вопросы в общем чате
6. Ожидать следующий вебинар