



Ярмарка
Мастеров
livemaster.ru

RНР - Вводная лекция

Ананьев Роман, Даниил Шевчук
Ведущий системный программист

Цель

- Основы языка PHP

Темы:

- версии
- операторы
- циклы
- типизация
- типы данных
- переменные

Наш язык PHP



Ярмарка
Мастеров
livemaster.ru

version 7.4 (но была и меньше), так что не забываем про php.net

php.net/manual/ru/function.json-encode.php

MySQL ★ Bookmarks PHPnet PHP.SU htmlbook jQuery Я Я M G W Вики livemaster.ru Ярмарка Wrike Хабр Sphinx/PHP.RU M:

php

Downloads

Documentation

Get Involved

Help

Список изменений

Версия	Описание
7.3.0	Добавлена константа <code>JSON_THROW_ON_ERROR</code> для параметра <code>options</code> .
7.2.0	Добавлены константы <code>JSON_INVALID_UTF8_IGNORE</code> и <code>JSON_INVALID_UTF8_SUBSTITUTE</code> для параметра <code>options</code> .
7.1.0	Добавлена константа <code>JSON_UNESCAPED_LINE_TERMINATORS</code> для параметра <code>options</code> .
7.1.0	При кодировании чисел с плавающей запятой используется serialize_precision вместо precision .



Переменные

Начинаются с \$ и буквы или _ (не с цифры)

Чувствительны к регистру

Не использовать зарезервированные переменные(например \$argc, \$_GET)

Область видимости (global)



Типы данных

Скалярные:

boolean

integer

float

string

Смешанные:

array

object

callable

iterable(c php 7.1)

Специальные:

resource

NULL

псевдотипы:

mixed

number

array|object

void

IF... else ... elseif



Ярмарка
Мастеров
livemaster.ru

Условные конструкции позволяют направлять работу программы в зависимости от условия по одному из возможных путей. И одной из таких конструкций в языке PHP является конструкция if..else

```
<?php
$a = 4;
if($a > 0){
    echo "Переменная а больше нуля";
}
else{
    echo "Переменная а меньше нуля";
}
echo "<br>конец выполнения программы";
?>
```

Операторы



Ярмарка
Мастеров
livemaster.ru

Операторы сравнения: ==, ===, !=, !==, >, <, >=, <=, <> , < = >

Логические операторы: !, &&, ||, and , or

Приоритет оператора <https://www.php.net/manual/ru/language.operators.precedence.php>

Вот потому важно использовать регламент написания PHP кода

Массивы

Ранее мы рассмотрели, как в переменные можно сохранить одиночное значение, например, одно число или одну строку. Но кроме того, мы можем сохранить в переменную набор значений. И для этого используются **массивы**.

```
$numbers = [1, 2, 3, 4];
```

```
$numbers = array(1, 2, 3, 4);
```


Ассоциативные массивы

Ассоциативные массивы представляют подвид массивов, в которых, в отличие от обычных массивов, в качестве ключа применяются строки.

```
$words = array("red" => "красный", "blue" => "синий", "green" => "зеленый");
```

```
$words = ["red" => "красный", "blue" => "синий", "green" => "зеленый"];
```

Глобальные массивы

\$GLOBALS Массив содержит ссылки на все переменные, объявленные в данном скрипте. Это ассоциативный массив, в котором имена переменных являются ключами.

\$_SERVER Массив содержит все данные о настройках среды выполнения скрипта и параметры сервера.

\$_GET Список переменных, переданных скрипту методом GET, т.е. через параметры URL-запроса.

\$_POST Список переменных, переданных скрипту методом POST.

\$_COOKIE Массив содержит все cookies, которые сервер установил на стороне пользователя.

\$_FILES Содержит список файлов, загруженных на сервер из формы. Более детально мы рассмотрим этот массив в уроке, посвящённом загрузке файлов на сервер.

\$_REQUEST Этот массив объединяет массивы \$GET, \$POST и \$COOKIE. очень часто бывает удобен при обработке пользовательских запросов, но применять его для защищённой обработки данных не стоит.

\$_SESSION Массив содержит все переменные сессии текущего пользователя.

Циклы

for

foreach

while

do ... while

Помним:

break - останавливает,

continue - пропускает.

```
$i = 0;
while ($i++ < 5) {
    while (1) {
        echo "вывод<br />\n";
        continue(2);
    }
    echo "Это никогда не будет выведено.<br />\n";
}
```

for

```
$cntArr = count($data);  
for ($i=0; $i<$cntArr; $i++) {  
    //полезный код  
}
```

```
for(;;){}
```

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

foreach



Ярмарка
Мастеров
livemaster.ru

```
//обычное использование  
foreach ($items as $rowItem) {  
    //полезное вычисление  
}
```

```
foreach ($items as $objectId => &$rowItem) {  
    //полезное вычисление  
    //можно делать так  
    $items[$objectId] = $rowItem;  
}  
unset($rowItem);
```

```
$array = [1,2,3,4];  
foreach( $array as $value ){  
    echo $value."\n";  
}
```

```
$array = [1,2,3,4];  
foreach( $array as $key => $value ){  
    echo $key."\t=>\t".$value."\n";  
}
```

while



Ярмарка
Мастеров
livemaster.ru

```
$count = 0;
while ($data = $oClient->pop() && $count < 5) {
    $pack[] = $data;
    $count++;
    if (sizeof($pack) >= $limit) {
        break;
    }
}
```

```
while(true) {}
```

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

do ... while



Ярмарка
Мастеров
livemaster.ru

```
do {  
    curl_multi_exec($mh, $running);  
    //код отправки  
} while($running > 0);
```

```
$i = 4;  
do {  
    echo $i;  
    $i--;  
} while ($i > 0);
```

Функции

Функции представляют собой блок инструкций, которые многократно можно вызывать в различных частях программы. Функции позволяют разделять программу на меньшие функциональные части.

```
function имя_функции([параметр [, ...]])  
{  
    // Инструкции  
}
```


Параметры функции

С помощью параметров мы можем передавать в функцию некоторые данные. Параметры определяются в скобках после названия функции как обычные переменные, отделенные друг от друга запятой.

```
function hello($name)
{
    echo "<h2>Hello $name</h2>";
}
```

```
hello("Tom");
```

Задание



1. Создать ассоциативный массив товаров где ключом будем артикул товара, а значением его цена.
 2. Вывести содержимое массив списком на странице.
-
1. Создать функцию `getProductDiscont()` которая с качестве аргумента принимает артикул товара и возвращает десять процентов от цены товара

Спасибо за внимание

Вопросы?