Что такое PHP?

PHP (рекурсивный акроним словосочетания *PHP: Hypertext Preprocessor*) - это распространенный язык программирования общего назначения с открытым исходным кодом. PHP сконструирован специально для ведения Web-разработок и его код может внедряться непосредственно в HTML.

Простой ответ, но что он может означать? Вот пример кода:

Пример #1 Пример программирования на PHP

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
    "http://www.w3.org/TR/html4/loose.dtd">  
<html>  
    <head>  
        <title>Пример</title>  
    </head>  
    <body>  
  
        <?php  
        echo "Привет, я - скрипт PHP!";  
        ?>  
  
    </body>  
</html>

Вместо рутинного вывода HTML-кода командами языка (как это происходит, например, в Perl или C), скрипт PHP содержит HTML с вкраплениями кода (в нашем случае, это вывод текста "Привет, я - скрипт PHP!"). Код PHP отделяется специальными [начальным и конечным тегами <?php и ?>](http://php.net/manual/ru/language.basic-syntax.phpmode.php), которые позволяют "переключаться" в "PHP-режим" и выходить из него.

PHP отличается от JavaScript тем, что PHP-скрипты выполняются на сервере и генерируют HTML, который посылается клиенту. Если бы у вас на сервере был размещен скрипт, подобный вышеприведенному, клиент получил бы только результат его выполнения, но не смог бы выяснить, какой именно код его произвел. Вы даже можете настроить свой сервер таким образом, чтобы обычные HTML-файлы обрабатывались процессором PHP, так что клиенты даже не смогут узнать, получают ли они обычный HTML-файл или результат выполнения скрипта.

Типы данных в PHP

PHP является языком программирования с [динамической типизацией](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%BD%D0%B0%D0%BC%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F), не требующим указания типа при объявлении переменных, равно как и самого объявления переменных. Преобразования между скалярными типами зачастую осуществляются неявно без дополнительных усилий (впрочем, PHP предоставляет широкие возможности и для явного преобразования типов).

К скалярным типам данных относятся:

* [целый тип](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF) (integer),
* [вещественный тип данных](https://ru.wikipedia.org/wiki/%D0%A7%D0%B8%D1%81%D0%BB%D0%B0_%D1%81_%D0%BF%D0%BB%D0%B0%D0%B2%D0%B0%D1%8E%D1%89%D0%B5%D0%B9_%D0%B7%D0%B0%D0%BF%D1%8F%D1%82%D0%BE%D0%B9) (float, double),
* [логический тип](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%82%D0%B8%D0%BF) (boolean),
* [строковый тип](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D1%80%D0%BE%D0%BA%D0%BE%D0%B2%D1%8B%D0%B9_%D1%82%D0%B8%D0%BF) (string),
* и специальный тип NULL.

К нескалярным типам относятся:

* «ресурс» (resource),
* [массив](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2) (array),
* [объект](https://ru.wikipedia.org/wiki/%D0%9E%D0%B1%D1%8A%D0%B5%D0%BA%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) (object),

К псевдо типамотносятся:

* [mixed](https://ru.wikipedia.org/w/index.php?title=Mixed&action=edit&redlink=1) любой тип
* [number](https://ru.wikipedia.org/w/index.php?title=Number&action=edit&redlink=1) число (integer либо float)
* [callback](https://ru.wikipedia.org/wiki/Callback_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)) (string или [анонимная функция](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BC%D1%8B%D0%BA%D0%B0%D0%BD%D0%B8%D0%B5_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)))
* [void](https://ru.wikipedia.org/wiki/Void) отсутствие параметров

Диапазон целых чисел (integer) в PHP зависит от [платформы](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%B0) (обычно, это диапазон 32-битных знаковых целых чисел, то есть, от −2 147 483 648 до 2 147 483 647). Числа можно задавать в десятичной, [восьмеричной](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D1%81%D1%8C%D0%BC%D0%B5%D1%80%D0%B8%D1%87%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%81%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F)и [шестнадцатеричной](https://ru.wikipedia.org/wiki/%D0%A8%D0%B5%D1%81%D1%82%D0%BD%D0%B0%D0%B4%D1%86%D0%B0%D1%82%D0%B5%D1%80%D0%B8%D1%87%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D1%81%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D1%8F) системах счисления. Диапазон вещественных чисел (double) также зависит от платформы (для 32-битной архитектуры диапазон позволяет оперировать числами от ±1.7×10−308 до ±1.7×10+308).

PHP предоставляет разработчикам [логический тип](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%B3%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B8%D0%B9_%D1%82%D0%B8%D0%BF) (boolean), способный принимать только два значения TRUE («истина») и FALSE («ложь»). При преобразовании в логический тип число 0, пустая строка, ноль в строке «0», NULL и пустой массив считаются равными FALSE. Все остальные значения автоматически преобразуются в TRUE.

Специальный тип NULL предназначен для переменных без определённого значения. Единственным значением данного типа является константа NULL. Тип NULL принимают неинициализированные переменные, переменные инициализированные константой NULL, а также переменные, удалённые при помощи конструкции unset().

Ссылки на внешние ресурсы имеют тип «ресурс» (resource). Переменные данного типа, как правило, представляют собой [дескриптор](https://ru.wikipedia.org/wiki/%D0%94%D0%B5%D1%81%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D1%80_(%D1%8F%D0%B7%D1%8B%D0%BA%D0%B8_%D1%80%D0%B0%D0%B7%D0%BC%D0%B5%D1%82%D0%BA%D0%B8)), позволяющий управлять внешними объектами, такими как файлы, динамические изображения, результирующие таблицы базы данных и т. п.

[Массивы](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2) (array) поддерживают числовые и строковые ключи и являются [гетерогенными](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2#.D0.A1.D0.BF.D0.B5.D1.86.D0.B8.D1.84.D0.B8.D1.87.D0.B5.D1.81.D0.BA.D0.B8.D0.B5_.D1.82.D0.B8.D0.BF.D1.8B_.D0.BC.D0.B0.D1.81.D1.81.D0.B8.D0.B2.D0.BE.D0.B2). Массивы могут содержать значения любых типов, включая другие массивы. Порядок элементов и их ключей сохраняется. Не совсем корректно называть php-массивы массивами, на самом деле это, скорее всего, упорядоченный [хеш](https://ru.wikipedia.org/wiki/%D0%90%D1%81%D1%81%D0%BE%D1%86%D0%B8%D0%B0%D1%82%D0%B8%D0%B2%D0%BD%D1%8B%D0%B9_%D0%BC%D0%B0%D1%81%D1%81%D0%B8%D0%B2" \l "PHP" \o "Ассоциативный массив). Возможно неожиданное поведение при использовании цикла for со счетчиком вместо foreach. Так, например, при сортировке массива с численными индексами функциями из стандартной библиотеки, сортируются и ключи тоже.

Указатель на функцию в PHP может быть представлен замыканием или псевдотипом [callback](https://ru.wikipedia.org/wiki/Callback_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)" \o "Callback (программирование)). Замыкание доступно с версии 5.3 и в коде выглядит как простое определение функции, в которую можно явно пробросить значения из контекста, например:

function($args, $argsN) use ($ctxVar, $ctxVar1) { ... }

callback тип может быть представлен:

* строкой (интерпретируется как название функции);
* массивом где нулевой и первый элемент — строки (интерпретируется как название статической функции класса);
* массивом где нулевой элемент — объект, а первый — строка (интерпретируется как метод у объекта).
* Для проверки является ли значение вызываемым следует использовать is\_callable($var)

Массивы в языке PHP

На самом деле массив в PHP - это упорядоченное отображение, которое устанавливает соответствие между *значеним* и *ключом*. Этот тип оптимизирован в нескольких направлениях, поэтому вы можете использовать его как собственно массив, список (вектор), хэш-таблицу (являющуюся реализацией карты), словарь, коллекцию, стек, очередь и, возможно, что-то еще. Так как значением массива может быть другой массив PHP, можно также создавать деревья и многомерные массивы.

Объяснение этих структур данных выходит за рамки данного справочного руководства, но вы найдете как минимум один пример по каждой из них. За дополнительной информацией вы можете обратиться к соответствующей литературе по этой обширной теме.

Массив (тип [array](http://php.net/manual/ru/language.types.array.php)) может быть создан языковой конструкцией [array()](http://php.net/manual/ru/function.array.php). language construct. В качестве параметров она принимает любое количество разделенных запятыми пар key => value (ключ => значение).

array(

key => value,

key2 => value2,

key3 => value3,

...

)

Запятая после последнего элемента массива необязательна и может быть опущена. Обычно это делается для однострочных массивов, т.е. *array(1, 2)* предпочтительней *array(1, 2, )*. Для многострочных массивов с другой стороны обычно используется завершающая запятая, так как позволяет легче добавлять новые элементы в конец массива.

Пример #1: Простой массив

<?php  
$array = array(  
    "foo" => "bar",  
    "bar" => "foo",  
);  
  
// Начиная с PHP 5.4  
$array = [  
    "foo" => "bar",  
    "bar" => "foo",  
];  
?>

Пример #2 Пример преобразования типов и перезаписи элементов

<?php  
$array = array(  
    1    => "a",  
    "1"  => "b",  
    1.5  => "c",  
    true => "d",  
);  
var\_dump($array);  
?>

Результат выполнения данного примера:

array(1) {

[1]=>

string(1) "d"

}

Пример #3 Смешанные ключи типов [integer](http://php.net/manual/ru/language.types.integer.php) и [string](http://php.net/manual/ru/language.types.string.php)

<?php  
$array = array(  
    "foo" => "bar",  
    "bar" => "foo",  
    100   => -100,  
    -100  => 100,  
);  
var\_dump($array);  
?>

Результат выполнения данного примера:

array(4) {

["foo"]=>

string(3) "bar"

["bar"]=>

string(3) "foo"

[100]=>

int(-100)

[-100]=>

int(100)

}

Пример #4 Индексированные массивы без ключа

<?php  
$array = array("foo", "bar", "hello", "world");  
var\_dump($array);  
?>

Результат выполнения данного примера:

array(4) {

[0]=>

string(3) "foo"

[1]=>

string(3) "bar"

[2]=>

string(5) "hello"

[3]=>

string(5) "world"

}

Пример #5 Ключи для некоторых элементов

<?php  
$array = array(  
         "a",  
         "b",  
    6 => "c",  
         "d",  
);  
var\_dump($array);  
?>

Результат выполнения данного примера:

array(4) {

[0]=>

string(1) "a"

[1]=>

string(1) "b"

[6]=>

string(1) "c"

[7]=>

string(1) "d"

}

Литература

1. Что такое PHP <http://php.net/manual/ru/intro-whatis.php>
2. Типы данных в PHP https://ru.wikipedia.org/wiki/PHP