Getting Started
    Introduction
    A simple tutorial
Language Reference
    Basic syntax
    Types
    Variables
    Constants
    Expressions
    Operators
    Control Structures
    Functions
    Classes and Objects
    Namespaces
    Enumerations
    Errors
    Exceptions
    Fibers
    Generators
    Attributes
    References Explained
    Predefined Variables
    Predefined Exceptions
    Predefined Interfaces and Classes
    Predefined Attributes
    Context options and parameters
    Supported Protocols and Wrappers

Security
    Introduction
    General considerations
    Installed as CGI binary
    Installed as an Apache module
    Session Security
    Filesystem Security
    Database Security
    Error Reporting
    User Submitted Data
    Hiding PHP
    Keeping Current
Features
    HTTP authentication with PHP
    Cookies
    Sessions
    Dealing with XForms
    Handling file uploads
    Using remote files
    Connection handling
    Persistent Database Connections
    Command line usage

Garbage Collection
DTrace Dynamic Tracing

Function Reference
Affecting PHP's Behaviour
Audio Formats Manipulation
Authentication Services
Command Line Specific Extensions
Compression and Archive Extensions
Cryptography Extensions
Database Extensions
Date and Time Related Extensions
File System Related Extensions
Human Language and Character Encoding Support
Image Processing and Generation
Mail Related Extensions
Mathematical Extensions
Non-Text MIME Output
Process Control Extensions
Other Basic Extensions
Other Services
Search Engine Extensions
Server Specific Extensions
Session Extensions
Text Processing
Variable and Type Related Extensions
Web Services
Windows Only Extensions
XML Manipulation
GUI Extensions

Keyboard Shortcuts

?	This help
j	Next menu item
k	Previous menu item
g p	Previous man page
g n	Next man page
G	Scroll to bottom
g g	Scroll to top
g h	Goto homepage
g s	Goto search (current page)
/	Focus search box

is_numeric »
« is_long

is_null

(PHP 4 >= 4.0.4, PHP 5, PHP 7, PHP 8)

is_null — Проверяет, равно ли значение переменной **`null`**

Описание

is_null(mixed `$value`): bool

Проверяет, равно ли значение переменной **`null`**.

Список параметров

`value`

Проверяемая переменная.

Возвращаемые значения

Возвращает **`true`**, если значение `value` — null, иначе **`false`**.

Примеры

Пример #1 Пример использования функции is_null()

```
<?php

error_reporting(E_ALL);

$foo = NULL;
var_dump(is_null($inexistent), is_null($foo));

?>
```

```
Notice: Undefined variable: inexistent in ...
bool(true)
bool(true)
```

Смотрите также

- Тип **`null`**
- isset() - Определяет, была ли установлена переменная значением, отличным от null
- is_bool() - Проверяет, представляет ли собой переменная логическое значение
- is_numeric() - Проверяет, содержит ли переменная число или числовую строку
- is_float() - Проверяет, представляет ли собой переменная число с плавающей точкой
- is_int() - Проверяет, представляет ли собой переменная целое число
- is_string() - Проверяет, представляет ли собой тип переменной строку
- is_object() - Проверяет, представляет ли собой переменная объект
- is_array() - Определяет, представляет ли собой переменная массив

+ add a note

User Contributed Notes 9 notes

up
down
311
***Malfist* ¶**
15 years ago

```
Micro optimization isn't worth it.
```

```
You had to do it ten million times to notice a difference, a little more than 2 seconds

$a===NULL; Took: 1.2424390316s
is_null($a); Took: 3.70693397522s

difference = 2.46449494362
difference/10,000,000 = 0.000000246449494362

The execution time difference between ===NULL and is_null is less than 250 nanoseconds. Go optimize something that matters.
```

up
down
160
george at fauxpanels dot com ¶
15 years ago

```
See how php parses different values. $var is the variable.

$var = NULL "" 0 "0" 1

strlen($var) = 0 0 1 1 1
is_null($var) = TRUE FALSE FALSE FALSE FALSE
$var == "" = TRUE TRUE TRUE FALSE FALSE
!$var = TRUE TRUE TRUE TRUE FALSE
!is_null($var) = FALSE TRUE TRUE TRUE TRUE
$var != "" = FALSE FALSE FALSE TRUE TRUE
$var = FALSE FALSE FALSE FALSE TRUE

Peace!
```

up
down
82
contact dot 01834e2c at renegade334 dot me dot uk ¶
8 years ago

```
In PHP 7 (phpng), is_null is actually marginally faster than ===, although the performance difference between the two is far smaller.

PHP 5.5.9
is_null - float(2.2381200790405)
=== - float(1.0024659633636)
=== faster by ~100ns per call

PHP 7.0.0-dev (built: May 19 2015 10:16:06)
is_null - float(1.4121870994568)
=== - float(1.4577329158783)
is_null faster by ~5ns per call
```

up
down
8
ahamilton9 ¶
1 year ago

```
A quick test in 2022 on PHP 8.1 confirms there is still no need to micro-optimize NULL checks:

<?php

// Comparison Operator
$before = microtime(true);
$var = null;
for ($i=0 ; $i<1000000000 ; $i++) {
if($var === null) {}
}
$after = microtime(true);
```

```
echo ' ===: ' . ($after - $before) . " seconds\n";

// Function
$before = microtime(true);
$var = null;
for ($i=0 ; $i<1000000000 ; $i++) {
if(is_null($var)) {}
}
$after = microtime(true);
echo 'is_null: ' . ($after - $before) . " seconds\n";

// ===: 4.1487579345703 seconds
// is_null: 4.1316878795624 seconds
```

up
down
13
ai dot unstmann at combase dot de ¶
16 years ago

For what I realized is that is_null($var) returns exactly the opposite of isset($var) , except that is_null($var) throws a notice if $var hasn't been set yet.

the following will prove that:

```
<?php

$quirks = array(null, true, false, 0, 1, '', "\0", "unset");

foreach($quirks as $var) {
if ($var === "unset") unset($var);

echo is_null($var) ? 1 : 0;
echo isset($var) ? 1 : 0;
echo "\n";
}

?>
```

this will print out something like:

```
10 // null
01 // true
01 // false
01 // 0
01 // 1
01 // ''
01 // "\0"
Notice: Undefined variable: var in /srv/www/htdocs/sandbox/null/nulltest.php on line 8
10 // (unset)
```

For the major quirky types/values is_null($var) obviously always returns the opposite of isset($var), and the notice clearly points out the faulty line with the is_null() statement. You might want to examine the return value of those functions in detail, but since both are specified to return boolean types there should be no doubt.

A second look into the PHP specs tells that is_null() checks whether a value is null or not. So, you may pass any VALUE to it, eg. the result of a function.
isset() on the other hand is supposed to check for a VARIABLE's existence, which makes it a language construct rather than a function. Its sole porpuse lies in that checking. Passing anything else will result in an error.

Knowing that, allows us to draw the following unlikely conclusion:

isset() as a language construct is way faster, more reliable and powerful than is_null() and should be prefered over is_null(), except for when you're directly passing a function's result, which is considered bad programming practice

anyways.
up
down
-6
normadize (a) gmail (d) com ¶
11 years ago
Using === NULL instead of is_null(), is actually useful in loaded server scenarios where you have hundreds or thousands of requests per second. Saving microseconds on a lot of "simple" operations in the entire PHP execution chain usually results in being able to serve more pages per second at the same speed, or lowering your cpu usage. People usually write very bad and slow code.
up
down
-7
strrev xc.noxeh@ellij ¶
15 years ago
```
$var===NULL is much faster than is_null($var) (with the same result)

I did some benchmarking with 10 million iterations:

$a=null;
isset($a); Took: 1.71841216087s
$a==NULL; Took: 1.27205181122s
$a===NULL; Took: 1.2424390316s
is_null($a); Took: 3.70693397522s
$a=5;
isset($a); Took: 1.15165400505s
$a==NULL; Took: 1.41901302338s
$a===NULL; Took: 1.21655392647s
is_null($a); Took: 3.78501200676s
error_reporting(E_ALL&~E_NOTICE);
unset($a);
isset($a); Took: 1.51441502571s
$a==NULL; Took: 16.5414860249s
$a===NULL; Took: 16.1273870468s
is_null($a); Took: 23.1918480396s

Please note, that isset is only included because it gives a good performance in any case; HOWEVER isset is NOT the same, or the opposite.
But you might be able to use isset() instead of null-checking.

You should not use is_null, except when you need a callback-function, or for conformity with is_int, is_float, etc.
```
up
down
-3
etimjoshua4 at gmail dot com ¶
3 years ago
```
I've found that for HTTP requests such as POST, is_null isn't the most reliable choice for checking if empty.

Try using

if(trim($var) == ""){
// do something
}

instead.

I think the request does something to the input that makes it definitely not NULL.
```
up
down
-40
michael at cannonbose dot com ¶
20 years ago

```
Regarding avoidance of NULLs in your MySQL queries, why not use IS NULL and IS NOT NULL in your WHERE clauses.

SELECT *
FROM someDatabase
WHERE someAttribute IS NOT NULL

Cheers,

Michael
```
+ add a note

- Функции для работы с переменными
  - boolval
  - debug_zval_dump
  - doubleval
  - empty
  - floatval
  - get_debug_type
  - get_defined_vars
  - get_resource_id
  - get_resource_type
  - gettype
  - intval
  - is_array
  - is_bool
  - is_callable
  - is_countable
  - is_double
  - is_float
  - is_int
  - is_integer
  - is_iterable
  - is_long
  - is_null
  - is_numeric
  - is_object
  - is_real
  - is_resource
  - is_scalar
  - is_string
  - isset
  - print_r
  - serialize
  - settype
  - strval
  - unserialize
  - unset
  - var_dump
  - var_export