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

Магические константы »
« Синтаксис

- Руководство по PHP
- Справочник языка
- Константы

Change language: Russian

Предопределённые константы

PHP предоставляет большой список предопределённых констант для каждого выполняемого скрипта. Многие из этих констант определяются различными модулями и будут присутствовать только в том случае, если эти модули доступны в результате динамической загрузки или в результате статической сборки.

+ add a note

User Contributed Notes 5 notes

up
down
291
vijaykoul_007 at rediffmail dot com ¶
18 years ago

```
the difference between
__FUNCTION__ and __METHOD__ as in PHP 5.0.4 is that

__FUNCTION__ returns only the name of the function

while as __METHOD__ returns the name of the class alongwith the name of the function

class trick
{
function doit()
{
echo __FUNCTION__;
}
function doitagain()
{
echo __METHOD__;
}
}
$obj=new trick();
$obj->doit();
output will be ---- doit
$obj->doitagain();
output will be ----- trick::doitagain
```

up
down
47
Tomek Perlak [tomekperlak at tlen pl] ¶
17 years ago

```
The __CLASS__ magic constant nicely complements the get_class() function.

Sometimes you need to know both:
- name of the inherited class
- name of the class actually executed

Here's an example that shows the possible solution:

<?php

class base_class
{
function say_a()
{
echo "'a' - said the " . __CLASS__ . "<br/>";
}
function say_b()
{
echo "'b' - said the " . get_class($this) . "<br/>";
}

}

class derived_class extends base_class
{
function say_a()
{
parent::say_a();
echo "'a' - said the " . __CLASS__ . "<br/>";
}

function say_b()
{
parent::say_b();
echo "'b' - said the " . get_class($this) . "<br/>";
}
}

$obj_b = new derived_class();

$obj_b->say_a();
echo "<br/>";
$obj_b->say_b();

?>

The output should look roughly like this:

'a' - said the base_class
'a' - said the derived_class

'b' - said the derived_class
'b' - said the derived_class
```

up
down
10
php at kenman dot net ¶
9 years ago

```
Just learned an interesting tidbit regarding __FILE__ and the newer __DIR__ with respect to code run from a network share:
the constants will return the *share* path when executed from the context of the share.

Examples:

// normal context
// called as "php -f c:\test.php"
__DIR__ === 'c:\';
__FILE__ === 'c:\test.php';

// network share context
// called as "php -f \\computerName\c$\test.php"
__DIR__ === '\\computerName\c$';
__FILE__ === '\\computerName\c$\test.php';

NOTE: realpath('.') always seems to return an actual filesystem path regardless of the execution context.
```

up
down
6
Sbastien Fauvel ¶

7 years ago

```
Note a small inconsistency when using __CLASS__ and __METHOD__ in traits (stand php 7.0.4): While __CLASS__ is working as advertized and returns dynamically the name of the class the trait is being used in, __METHOD__ will actually prepend the trait name instead of the class name!
```

up
down
0
public at taliesinnuin dot net ¶
3 years ago

```
If you're using PHP with fpm (common in this day and age), be aware that __DIR__ and __FILE__ will return values based on the fpm root which MAY differ from its actual location on the file system.

This can cause temporary head-scratching if deploying an app where php files within the web root pull in PHP files from outside of itself (a very common case). You may be wondering why __DIR__ returns "/" when the file itself lives in /var/www/html or whathaveyou.

You might handle such a situation by having NGINX explicitly add the necessary part of the path in its fastcgi request and then you can set the root on the FPM process / server / container to be something other than the webroot (so long as no other way it could become publicly accessible).

Hope that saves someone five minutes who's moving code to FPM that uses __DIR__.
```

+add a note