Search

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

Псевдонимирование и импорт »
« Пространства имён и динамические особенности языка

- Руководство по PHP
- Справочник языка
- Пространства имён

Change language: Russian

Ключевое слово namespace и магическая константа __NAMESPACE__

(PHP 5 >= 5.3.0, PHP 7, PHP 8)

PHP поддерживает два способа абстрактного доступа к элементам в текущем пространстве имён: магическая константа **__NAMESPACE__** и ключевое слово `namespace`.

Значение константы **__NAMESPACE__** — это строка, которая содержит имя текущего пространства имён. В глобальном пространстве, вне пространства имён, она содержит пустую строку.

Пример #1 Пример записи константы __NAMESPACE__ в коде с пространством имён

```
<?php

namespace MyProject;

echo '"', __NAMESPACE__, '"'; // Выводит «MyProject»

?>
```

Пример #2 Пример записи константы __NAMESPACE__ в глобальном пространстве

```
<?php

echo '"', __NAMESPACE__, '"'; // Выводит «»
?>
```

Константа **__NAMESPACE__** полезна для динамически конструируемых имён, например:

Пример #3 Константа __NAMESPACE__ и динамическое конструирование имени

```
<?php

namespace MyProject;

function get($classname)
{
$a = __NAMESPACE__ . '\\' . $classname;
return new $a;
}

?>
```

Ключевое слово `namespace` разрешено указывать для явного запроса элемента из текущего пространства имён или из подпространства. Это эквивалент ключевого слова `self` для классов в пространстве имён.

Пример #4 Ключевое слово namespace внутри пространства имён

```
<?php

namespace MyProject;

use blah\blah as mine; // Смотрите «Пространства имён: псевдонимирование и импорт»

blah\mine(); // Вызывает функцию MyProject\blah\mine()
namespace\blah\mine(); // Вызывает функцию MyProject\blah\mine()

namespace\func(); // Вызывает функцию MyProject\func()
namespace\sub\func(); // Вызывает функцию MyProject\sub\func()
namespace\cname::method(); // Вызывает статический метод method класса MyProject\cname
$a = new namespace\sub\cname(); // Создаёт экземпляр класса MyProject\sub\cname
$b = namespace\CONSTANT; // Присваивает значение константы MyProject\CONSTANT переменной $b

?>
```

Пример #5 Ключевое слово namespace в глобальном коде

```php
<?php

namespace\func(); // Вызывает функцию func()
namespace\sub\func(); // Вызывает функцию sub\func()
namespace\cname::method(); // Вызывает статический метод method класса cname
$a = new namespace\sub\cname(); // Создаёт экземпляр класса sub\cname
$b = namespace\CONSTANT; // Присваивает значение константы CONSTANT переменной $b

?>
```


User Contributed Notes 3 notes

85
a dot schaffhirt at sedna-soft dot de ¶
14 years ago
Just in case you wonder what the practical use of the namespace keyword is...

It can explicitly refer to classes from the current namespace regardless of possibly "use"d classes with the same name from other namespaces. However, this does not apply for functions.

Example:

```
<?php
namespace foo;
class Xyz {}
function abc () {}
?>

<?php
namespace bar;
class Xyz {}
function abc () {}
?>

<?php
namespace bar;
use foo\Xyz;
use foo\abc;
new Xyz(); // instantiates \foo\Xyz
new namespace\Xyz(); // instantiates \bar\Xyz
abc(); // invokes \bar\abc regardless of the second use statement
\foo\abc(); // it has to be invoked using the fully qualified name
?>
```

Hope, this can save someone from some trouble.

Best regards.

0
bharatthapa45 at gmail dot com ¶
1 year ago
Difference between __NAMESPACE__ and keyword 'namespace' that I find relevant is when invoking a class:

```
<?php
namespace MyApp;
```

```
class App {
static function app(){
echo 'hello app';
}
}

// this will work:
$obj = new namespace\App::app();

// this will not work
$obj = new __NAMESPACE__\App::app();

// however this will work:
$obj = __NAMESPACE__ . '\App';
$obj::foo();

?>
```

up
down
-15
cornichonche at gmail dot com ¶
5 years ago

```
The example 4 is wrong.
Using php 7.2

<?php
namespace monProjet;

use function blah\blah as mine;

blah\mine(); // Will NOT work
mine(); // Will work
?>
```

＋add a note