- 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

Обзор »

« Журнал изменений ООП

- Руководство по PHP
- Справочник языка

Change language: Russian

Submit a Pull Request Report a Bug

Пространства имён

Содержание

+ add a note

User Contributed Notes 4 notes

up
down
249
***Anonymous* ¶**
12 years ago

```
The keyword 'use' has two different applications, but the reserved word table links to here.

It can apply to namespace constucts:

file1:
<?php namespace foo;
class Cat {
static function says() {echo 'meoow';} } ?>

file2:
<?php namespace bar;
class Dog {
static function says() {echo 'ruff';} } ?>

file3:
<?php namespace animate;
class Animal {
static function breathes() {echo 'air';} } ?>

file4:
<?php namespace fub;
include 'file1.php';
include 'file2.php';
include 'file3.php';
use foo as feline;
use bar as canine;
use animate;
echo \feline\Cat::says(), "<br />\n";
echo \canine\Dog::says(), "<br />\n";
echo \animate\Animal::breathes(), "<br />\n"; ?>

Note that
felineCat::says()
should be
\feline\Cat::says()
```

(and similar for the others)
but this comment form deletes the backslash (why???)

The 'use' keyword also applies to closure constructs:

```
<?php function getTotal($products_costs, $tax)
{
$total = 0.00;

$callback =
function ($pricePerItem) use ($tax, &$total)
{

$total += $pricePerItem * ($tax + 1.0);
};

array_walk($products_costs, $callback);
return round($total, 2);
}
?>
```

up
down
26

Anonymous ¶

7 years ago

Tested on PHP 7.0.5, Windows
The line "use animate;" equals the line "use animate as animate;"
but the "use other\animate;" equals "use other\animate as animate;"

```
file1:
<?php namespace foo;
class Cat {
static function says() {echo 'meoow';} } ?>

file2:
<?php namespace bar;
class Dog {
static function says() {echo 'ruff';} } ?>

file3:
<?php namespace other\animate;
class Animal {
static function breathes() {echo 'air';} } ?>

file4:
<?php namespace fub;
include 'file1.php';
include 'file2.php';
include 'file3.php';
use foo as feline;
use bar as canine;
use other\animate; //use other\animate as animate;
echo feline\Cat::says(), "<br />\n";
echo canine\Dog::says(), "<br />\n";
echo \animate\Animal::breathes(), "<br />\n"; ?>
```

up
down
10

davidkennedy85 at gmail dot com ¶

8 years ago

In addition to using namespaces and closures, the use keyword has another new meaning as of PHP 5.4 - using traits:

```php
<?php
trait Hello {
public function sayHello() {
echo 'Hello ';
}
}

trait World {
public function sayWorld() {
echo 'World';
}
}

class MyHelloWorld {
use Hello, World;
public function sayExclamationMark() {
echo '!';
}
}

$o = new MyHelloWorld();
$o->sayHello();
$o->sayWorld();
$o->sayExclamationMark();
?>
```

More info here: http://php.net/manual/en/language.oop5.traits.php

[up](#)

[down](#)

8

[varuninorbit at yahoo dot co dot in ¶](#)

8 years ago

here is a simple example to use namespace

```php
<?php

namespace app\a{
class one{
public static function _1(){
echo 'a one _1<br>';
}
}
}

namespace app\b{
class one{
public static function _2(){
echo 'b one _2<br>';
}
}
}

namespace app{

echo a\one::_1();
echo b\one::_2();
echo a\two::_1();
}

namespace app\a{
class two{
public static function _1(){
```

```
echo 'a two _1<br>';
}
}
}

prints
a one _1
b one _2
a two _1
```

+ add a note

- Справочник языка
  - Основы синтаксиса
  - Типы
  - Переменные
  - Константы
  - Выражения
  - Операторы
  - Управляющие конструкции
  - Функции
  - Классы и объекты
  - Пространства имён
  - Перечисления
  - Ошибки
  - Исключения
  - Fibers
  - Генераторы
  - Атрибуты
  - Объяснение ссылок
  - Предопределённые переменные
  - Предопределённые исключения
  - Встроенные интерфейсы и классы
  - Предопределённые атрибуты
  - Контекстные опции и параметры
  - Поддерживаемые протоколы и обёртки