# **PHP中spl\_autoload\_register()函数用法实例详解**

这篇文章主要介绍了PHP中spl\_autoload\_register()函数用法,结合实例形式分析了\_\_autoload函数及spl\_autoload\_register函数的相关使用技巧,需要的朋友可以参考下

本文实例分析了PHP中spl\_autoload\_register()函数用法。分享给大家供大家参考，具体如下：

在了解这个函数之前先来看另一个函数：\_\_autoload。

**一、\_\_autoload**

这是一个自动加载函数，在PHP5中，当我们实例化一个未定义的类时，就会触发此函数。看下面例子：

printit.class.php：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | <?php  **class** PRINTIT {  **function** doPrint() {   echo 'hello world';   }  }  ?> |

index.php

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | <?  **function** \_\_autoload( $class ) {   $file = $class . '.class.php';  **if** ( is\_file($file) ) {  **require\_once**($file);   }  }  $obj = **new** PRINTIT();  $obj->doPrint();?> |

运行index.php后正常输出hello world。在index.php中，由于没有包含printit.class.php，在实例化printit时，自动调用\_\_autoload函数，参数$class的值即为类名printit，此时printit.class.php就被引进来了。

在面向对象中这种方法经常使用，可以避免书写过多的引用文件，同时也使整个系统更加灵活。

**二、spl\_autoload\_register()**

再看spl\_autoload\_register()，这个函数与\_\_autoload有与曲同工之妙，看个简单的例子：

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | <?  **function** loadprint( $class ) {   $file = $class . '.class.php';  **if** (is\_file($file)) {  **require\_once**($file);   }  }  spl\_autoload\_register( 'loadprint' );  $obj = **new** PRINTIT();  $obj->doPrint();?> |

将\_\_autoload换成loadprint函数。但是loadprint不会像\_\_autoload自动触发，这时spl\_autoload\_register()就起作用了，它告诉PHP碰到没有定义的类就执行loadprint()。

spl\_autoload\_register() 调用静态方法

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | <?  **class** test {  **public** **static** **function** loadprint( $class ) {   $file = $class . '.class.php';  **if** (is\_file($file)) {  **require\_once**($file);   }   }  }  spl\_autoload\_register( **array**('test','loadprint') );  //另一种写法：spl\_autoload\_register( "test::loadprint" );  $obj = **new** PRINTIT();  $obj->doPrint();?> |

**spl\_autoload\_register**

(PHP 5 >= 5.1.2)

spl\_autoload\_register — 注册\_\_autoload()函数

**说明**

bool spl\_autoload\_register ([ callback $autoload\_function ] )  
将函数注册到SPL \_\_autoload函数栈中。如果该栈中的函数尚未激活，则激活它们。

如果在你的程序中已经实现了\_\_autoload函数，它必须显式注册到\_\_autoload栈中。因为spl\_autoload\_register()函数会将Zend Engine中的\_\_autoload函数取代为spl\_autoload() 或 spl\_autoload\_call()。

**参数**

autoload\_function

欲注册的自动装载函数。如果没有提供任何参数，则自动注册autoload的默认实现函数spl\_autoload()。

**返回值**

如果成功则返回 TRUE，失败则返回 FALSE。

注：SPL是Standard PHP Library(标准PHP库)的缩写。它是PHP5引入的一个扩展库，其主要功能包括autoload机制的实现及包括各种Iterator接口或类。SPL autoload机制的实现是通过将函数指针autoload\_func指向自己实现的具有自动装载功能的函数来实现的。SPL有两个不同的函数spl\_autoload, spl\_autoload\_call，通过将autoload\_func指向这两个不同的函数地址来实现不同的自动加载机制。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | classLOAD  {   staticfunctionloadClass($class\_name)    {      $filename= $class\_name.".class.php";   $path= "include/".$filename  **if**(is\_file($path)) returninclude$path;    }  }  /\*\*   \* 设置对象的自动载入   \* spl\_autoload\_register — Register given function as \_\_autoload() implementation   \*/  spl\_autoload\_register(**array**('LOAD', 'loadClass'));  /\*\*  \*\_\_autoload 方法在 spl\_autoload\_register 后会失效，因为 autoload\_func 函数指针已指向 spl\_autoload 方法  \* 可以通过下面的方法来把 \_autoload 方法加入 autoload\_functions list  \*/  spl\_autoload\_register( '\_\_autoload'); |

如果同时用spl\_autoload\_register注册了一个类的方法和\_\_autoload函数，那么，会根据注册的先后，如果在第一个注册的方法或函数里加载了类文件，就不会再执行第二个被注册的类的方法或函数。反之就会执行第二个被注册的类的方法或函数。

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41 | <?php  **class** autoloader {  **public** **static** $loader;  **public** **static** **function** init() {  **if** (self::$loader == NULL)        self::$loader = **new** self ();  **return** self::$loader;    }  **public** **function** \_\_construct() {      spl\_autoload\_register ( **array** ($this, 'model' ) );      spl\_autoload\_register ( **array** ($this, 'helper' ) );      spl\_autoload\_register ( **array** ($this, 'controller' ) );      spl\_autoload\_register ( **array** ($this, 'library' ) );    }  **public** **function** library($class) {      set\_include\_path ( get\_include\_path () . PATH\_SEPARATOR . '/lib/' );      spl\_autoload\_extensions ( '.library.php' );      spl\_autoload ( $class );    }  **public** **function** controller($class) {      $class = preg\_replace ( '/\_controller$/ui', '', $class );      set\_include\_path ( get\_include\_path () . PATH\_SEPARATOR . '/controller/' );      spl\_autoload\_extensions ( '.controller.php' );      spl\_autoload ( $class );    }  **public** **function** model($class) {      $class = preg\_replace ( '/\_model$/ui', '', $class );      set\_include\_path ( get\_include\_path () . PATH\_SEPARATOR . '/model/' );      spl\_autoload\_extensions ( '.model.php' );      spl\_autoload ( $class );    }  **public** **function** helper($class) {      $class = preg\_replace ( '/\_helper$/ui', '', $class );      set\_include\_path ( get\_include\_path () . PATH\_SEPARATOR . '/helper/' );      spl\_autoload\_extensions ( '.helper.php' );      spl\_autoload ( $class );    }  }  //call  autoloader::init ();  ?> |