[理解版](../../普通函数与回调函数的区别.docx)

自 PHP 5.4 起可用 [callable](https://www.php.net/manual/zh/language.types.callable.php) 类型指定回调类型 callback。本文档基于同样理由使用 [callback](https://www.php.net/manual/zh/language.pseudo-types.php#language.types.callback) 类型信息。

一些函数如 [call\_user\_func()](https://www.php.net/manual/zh/function.call-user-func.php) 或 [usort()](https://www.php.net/manual/zh/function.usort.php) 可以接受用户自定义的回调函数作为参数。回调函数不止可以是简单函数，还可以是对象的方法，包括静态类方法。

**传递**

PHP是将函数以[string](https://www.php.net/manual/zh/language.types.string.php)形式传递的。 可以使用任何内置或用户自定义函数，但除了语言结构例如：[array()](https://www.php.net/manual/zh/function.array.php)，[echo](https://www.php.net/manual/zh/function.echo.php)，[empty()](https://www.php.net/manual/zh/function.empty.php)，[eval()](https://www.php.net/manual/zh/function.eval.php)，[exit()](https://www.php.net/manual/zh/function.exit.php)，[isset()](https://www.php.net/manual/zh/function.isset.php)，[list()](https://www.php.net/manual/zh/function.list.php)，[print](https://www.php.net/manual/zh/function.print.php) 或 [unset()](https://www.php.net/manual/zh/function.unset.php)。

一个已实例化的 [object](https://www.php.net/manual/zh/language.types.object.php) 的方法被作为 [array](https://www.php.net/manual/zh/language.types.array.php) 传递，下标 0 包含该 [object](https://www.php.net/manual/zh/language.types.object.php)，下标 1 包含方法名。 在同一个类里可以访问 protected 和 private 方法。

静态类方法也可不经实例化该类的对象而传递，只要在下标 0 中包含类名而不是对象。自 PHP 5.2.3 起，也可以传递 *'ClassName::methodName'*。

除了普通的用户自定义函数外，也可传递 [匿名函数](https://www.php.net/manual/zh/functions.anonymous.php) 给回调参数。

**Example #1 回调函数示例**

<?php   
  
// An example callback function  
function my\_callback\_function() {  
    echo 'hello world!';  
}  
  
// An example callback method  
class MyClass {  
    static function myCallbackMethod() {  
        echo 'Hello World!';  
    }  
}  
  
// Type 1: Simple callback  
call\_user\_func('my\_callback\_function');   
  
// Type 2: Static class method call  
call\_user\_func(array('MyClass', 'myCallbackMethod'));   
  
// Type 3: Object method call  
$obj = new MyClass();  
call\_user\_func(array($obj, 'myCallbackMethod'));  
  
// Type 4: Static class method call (As of PHP 5.2.3)  
call\_user\_func('MyClass::myCallbackMethod');  
  
// Type 5: Relative static class method call (As of PHP 5.3.0)  
class A {  
    public static function who() {  
        echo "A\n";  
    }  
}  
  
class B extends A {  
    public static function who() {  
        echo "B\n";  
    }  
}  
  
call\_user\_func(array('B', 'parent::who')); // A  
  
// Type 6: Objects implementing \_\_invoke can be used as callables (since PHP 5.3)  
class C {  
    public function \_\_invoke($name) {  
        echo 'Hello ', $name, "\n";  
    }  
}  
  
$c = new C();  
call\_user\_func($c, 'PHP!');  
?>

**Example #2 使用 Closure 的示例**

<?php  
// Our closure  
$double = function($a) {  
    return $a \* 2;  
};  
  
// This is our range of numbers  
$numbers = range(1, 5);  
  
// Use the closure as a callback here to   
// double the size of each element in our   
// range  
$new\_numbers = array\_map($double, $numbers);  
  
print implode(' ', $new\_numbers);  
?>

以上例程会输出：

2 4 6 8 10

**Note**:

在函数中注册有多个回调内容时(如使用 [call\_user\_func()](https://www.php.net/manual/zh/function.call-user-func.php) 与 [call\_user\_func\_array()](https://www.php.net/manual/zh/function.call-user-func-array.php))，如在前一个回调中有未捕获的异常，其后的将不再被调用。