# 代码风格规范

本篇规范是 [PSR-1](https://github.com/PizzaLiu/PHP-FIG/blob/master/PSR-1-basic-coding-standard-cn.md) 基本代码规范的继承与扩展。

本规范希望通过制定一系列规范化PHP代码的规则，以减少在浏览不同作者的代码时，因代码风格的不同而造成不便。

当多名程序员在多个项目中合作时，就需要一个共同的编码规范， 而本文中的风格规范源自于多个不同项目代码风格的共同特性， 因此，本规范的价值在于我们都遵循这个编码风格，而不是在于它本身。

关键词 “必须”("MUST")、“一定不可/一定不能”("MUST NOT")、“需要”("REQUIRED")、 “将会”("SHALL")、“不会”("SHALL NOT")、“应该”("SHOULD")、“不该”("SHOULD NOT")、 “推荐”("RECOMMENDED")、“可以”("MAY")和”可选“("OPTIONAL")的详细描述可参见 [RFC 2119](http://www.ietf.org/rfc/rfc2119.txt) 。

## 1. 概览

* 代码**必须**遵循 [PSR-1](https://github.com/PizzaLiu/PHP-FIG/blob/master/PSR-1-basic-coding-standard-cn.md) 中的编码规范 。
* 代码**必须**使用4个空格符而不是 tab键 进行缩进。
* 每行的字符数**应该**软性保持在80个之内， 理论上**一定不可**多于120个， 但**一定不能**有硬性限制。
* 每个 namespace 命名空间声明语句和 use 声明语句块后面，**必须**插入一个空白行。
* 类的开始花括号({)**必须**写在函数声明后自成一行，结束花括号(})也**必须**写在函数主体后自成一行。
* 方法的开始花括号({)**必须**写在函数声明后自成一行，结束花括号(})也**必须**写在函数主体后自成一行。
* 类的属性和方法**必须**添加访问修饰符（private、protected 以及 public）， abstract 以及 final **必须**声明在访问修饰符之前，而 static **必须**声明在访问修饰符之后。
* 控制结构的关键字后**必须**要有一个空格符，而调用方法或函数时则**一定不能**有。
* 控制结构的开始花括号({)**必须**写在声明的同一行，而结束花括号(})**必须**写在主体后自成一行。
* 控制结构的开始左括号后和结束右括号前，都**一定不能**有空格符。

### 1.1. 例子

以下例子程序简单地展示了以上大部分规范：

<?php  
namespace Vendor\Package;  
  
use FooInterface;  
use BarClass as Bar;  
use OtherVendor\OtherPackage\BazClass;  
  
class Foo extends Bar implements FooInterface  
{  
 public function sampleFunction($a, $b = null)  
 {  
 if ($a === $b) {  
 bar();  
 } elseif ($a > $b) {  
 $foo->bar($arg1);  
 } else {  
 BazClass::bar($arg2, $arg3);  
 }  
 }  
  
 final public static function bar()  
 {  
 // method body  
 }  
}

## 2. 通则

### 2.1 基本编码准则

代码**必须**符合 [PSR-1](https://github.com/PizzaLiu/PHP-FIG/blob/master/PSR-1-basic-coding-standard-cn.md) 中的所有规范。

### 2.2 文件

所有PHP文件**必须**使用Unix LF (linefeed)作为行的结束符。

所有PHP文件**必须**以一个空白行作为结束。

纯PHP代码文件**必须**省略最后的 ?> 结束标签。

### 2.3. 行

行的长度**一定不能**有硬性的约束。

软性的长度约束**一定**要限制在120个字符以内，若超过此长度，带代码规范检查的编辑器**一定**要发出警告，不过**一定不可**发出错误提示。

每行**不应该**多于80个字符，大于80字符的行**应该**折成多行。

非空行后**一定不能**有多余的空格符。

空行**可以**使得阅读代码更加方便以及有助于代码的分块。

每行**一定不能**存在多于一条语句。

### 2.4. 缩进

代码**必须**使用4个空格符的缩进，**一定不能**用 tab键 。

备注: 使用空格而不是tab键缩进的好处在于， 避免在比较代码差异、打补丁、重阅代码以及注释时产生混淆。 并且，使用空格缩进，让对齐变得更方便。

### 2.5. 关键字 以及 True/False/Null

PHP所有 [关键字](http://php.net/manual/en/reserved.keywords.php)**必须**全部小写。

常量 true 、false 和 null 也**必须**全部小写。

## 3. namespace 以及 use 声明

namespace 声明后 必须 插入一个空白行。

所有 use 必须 在 namespace 后声明。

每条 use 声明语句 必须 只有一个 use 关键词。

use 声明语句块后 必须 要有一个空白行。

例如：

<?php  
namespace Vendor\Package;  
  
use FooClass;  
use BarClass as Bar;  
use OtherVendor\OtherPackage\BazClass;  
  
// ... additional PHP code ...

## 4. 类、属性和方法

此处的“类”泛指所有的class类、接口以及traits可复用代码块。

### 4.1. 扩展与继承

关键词 extends 和 implements**必须**写在类名称的同一行。

类的开始花括号**必须**独占一行，结束花括号也**必须**在类主体后独占一行。

<?php  
namespace Vendor\Package;  
  
use FooClass;  
use BarClass as Bar;  
use OtherVendor\OtherPackage\BazClass;  
  
class ClassName extends ParentClass implements \ArrayAccess, \Countable  
{  
 // constants, properties, methods  
}

implements 的继承列表也**可以**分成多行，这样的话，每个继承接口名称都**必须**分开独立成行，包括第一个。

<?php  
namespace Vendor\Package;  
  
use FooClass;  
use BarClass as Bar;  
use OtherVendor\OtherPackage\BazClass;  
  
class ClassName extends ParentClass implements  
 \ArrayAccess,  
 \Countable,  
 \Serializable  
{  
 // constants, properties, methods  
}

### 4.2. 属性

每个属性都**必须**添加访问修饰符。

**一定不可**使用关键字 var 声明一个属性。

每条语句**一定不可**定义超过一个属性。

**不要**使用下划线作为前缀，来区分属性是 protected 或 private。

以下是属性声明的一个范例：

<?php  
namespace Vendor\Package;  
  
class ClassName  
{  
 public $foo = null;  
}

### 4.3. 方法

所有方法都**必须**添加访问修饰符。

**不要**使用下划线作为前缀，来区分方法是 protected 或 private。

方法名称后**一定不能**有空格符，其开始花括号**必须**独占一行，结束花括号也**必须**在方法主体后单独成一行。参数左括号后和右括号前**一定不能**有空格。

一个标准的方法声明可参照以下范例，留意其括号、逗号、空格以及花括号的位置。

<?php  
namespace Vendor\Package;  
  
class ClassName  
{  
 public function fooBarBaz($arg1, &$arg2, $arg3 = [])  
 {  
 // method body  
 }  
}

### 4.4. 方法的参数

参数列表中，每个逗号后面**必须**要有一个空格，而逗号前面**一定不能**有空格。

有默认值的参数，**必须**放到参数列表的末尾。

<?php  
namespace Vendor\Package;  
  
class ClassName  
{  
 public function foo($arg1, &$arg2, $arg3 = [])  
 {  
 // method body  
 }  
}

参数列表**可以**分列成多行，这样，包括第一个参数在内的每个参数都**必须**单独成行。

拆分成多行的参数列表后，结束括号以及方法开始花括号 必须 写在同一行，中间用一个空格分隔。

<?php  
namespace Vendor\Package;  
  
class ClassName  
{  
 public function aVeryLongMethodName(  
 ClassTypeHint $arg1,  
 &$arg2,  
 array $arg3 = []  
 ) {  
 // method body  
 }  
}

### 4.5. abstract 、 final 、 以及 static

需要添加 abstract 或 final 声明时， **必须**写在访问修饰符前，而 static 则**必须**写在其后。

<?php  
namespace Vendor\Package;  
  
abstract class ClassName  
{  
 protected static $foo;  
  
 abstract protected function zim();  
  
 final public static function bar()  
 {  
 // method body  
 }  
}

### 4.6. 方法及函数调用

方法及函数调用时，方法名或函数名与参数左括号之间**一定不能**有空格，参数右括号前也 **一定不能**有空格。每个参数前**一定不能**有空格，但其后**必须**有一个空格。

<?php  
bar();  
$foo->bar($arg1);  
Foo::bar($arg2, $arg3);

参数**可以**分列成多行，此时包括第一个参数在内的每个参数都**必须**单独成行。

<?php  
$foo->bar(  
 $longArgument,  
 $longerArgument,  
 $muchLongerArgument  
);

## 5. 控制结构

控制结构的基本规范如下：

* 控制结构关键词后**必须**有一个空格。
* 左括号 ( 后**一定不能**有空格。
* 右括号 ) 前也**一定不**能有空格。
* 右括号 ) 与开始花括号 { 间**一定**有一个空格。
* 结构体主体**一定**要有一次缩进。
* 结束花括号 } **一定**在结构体主体后单独成行。

每个结构体的主体都**必须**被包含在成对的花括号之中， 这能让结构体更加结构话，以及减少加入新行时，出错的可能性。

### 5.1. if 、 elseif 和 else

标准的 if 结构如下代码所示，留意 括号、空格以及花括号的位置， 注意 else 和 elseif 都与前面的结束花括号在同一行。

<?php  
if ($expr1) {  
 // if body  
} elseif ($expr2) {  
 // elseif body  
} else {  
 // else body;  
}

**应该**使用关键词 elseif 代替所有 else if ，以使得所有的控制关键字都像是单独的一个词。

### 5.2. switch 和 case

标准的 switch 结构如下代码所示，留意括号、空格以及花括号的位置。 case 语句**必须**相对 switch 进行一次缩进，而 break 语句以及 case 内的其它语句都 必须 相对 case 进行一次缩进。 如果存在非空的 case 直穿语句，主体里必须有类似 // no break 的注释。

<?php  
switch ($expr) {  
 case 0:  
 echo 'First case, with a break';  
 break;  
 case 1:  
 echo 'Second case, which falls through';  
 // no break  
 case 2:  
 case 3:  
 case 4:  
 echo 'Third case, return instead of break';  
 return;  
 default:  
 echo 'Default case';  
 break;  
}

### 5.3. while 和 do while

一个规范的 while 语句应该如下所示，注意其 括号、空格以及花括号的位置。

<?php  
while ($expr) {  
 // structure body  
}

标准的 do while 语句如下所示，同样的，注意其 括号、空格以及花括号的位置。

<?php  
do {  
 // structure body;  
} while ($expr);

### 5.4. for

标准的 for 语句如下所示，注意其 括号、空格以及花括号的位置。

<?php  
for ($i = 0; $i < 10; $i++) {  
 // for body  
}

### 5.5. foreach

标准的 foreach 语句如下所示，注意其 括号、空格以及花括号的位置。

<?php  
foreach ($iterable as $key => $value) {  
 // foreach body  
}

### 5.6. try, catch

标准的 try catch 语句如下所示，注意其 括号、空格以及花括号的位置。

<?php  
try {  
 // try body  
} catch (FirstExceptionType $e) {  
 // catch body  
} catch (OtherExceptionType $e) {  
 // catch body  
}

## 6. 闭包

闭包声明时，关键词 function 后以及关键词 use 的前后都**必须**要有一个空格。

开始花括号**必须**写在声明的同一行，结束花括号**必须**紧跟主体结束的下一行。

参数列表和变量列表的左括号后以及右括号前，**必须不能**有空格。

参数和变量列表中，逗号前**必须不能**有空格，而逗号后**必须**要有空格。

闭包中有默认值的参数**必须**放到列表的后面。

标准的闭包声明语句如下所示，注意其 括号、逗号、空格以及花括号的位置。

<?php  
$closureWithArgs = function ($arg1, $arg2) {  
 // body  
};  
  
$closureWithArgsAndVars = function ($arg1, $arg2) use ($var1, $var2) {  
 // body  
};

参数列表以及变量列表**可以**分成多行，这样，包括第一个在内的每个参数或变量都**必须**单独成行，而列表的右括号与闭包的开始花括号**必须**放在同一行。

以下几个例子，包含了参数和变量列表被分成多行的多情况。

<?php  
$longArgs\_noVars = function (  
 $longArgument,  
 $longerArgument,  
 $muchLongerArgument  
) {  
 // body  
};  
  
$noArgs\_longVars = function () use (  
 $longVar1,  
 $longerVar2,  
 $muchLongerVar3  
) {  
 // body  
};  
  
$longArgs\_longVars = function (  
 $longArgument,  
 $longerArgument,  
 $muchLongerArgument  
) use (  
 $longVar1,  
 $longerVar2,  
 $muchLongerVar3  
) {  
 // body  
};  
  
$longArgs\_shortVars = function (  
 $longArgument,  
 $longerArgument,  
 $muchLongerArgument  
) use ($var1) {  
 // body  
};  
  
$shortArgs\_longVars = function ($arg) use (  
 $longVar1,  
 $longerVar2,  
 $muchLongerVar3  
) {  
 // body  
};

注意，闭包被直接用作函数或方法调用的参数时，以上规则仍然适用。

<?php  
$foo->bar(  
 $arg1,  
 function ($arg2) use ($var1) {  
 // body  
 },  
 $arg3  
);

## 7. 总结

以上规范难免有疏忽，其中包括但不仅限于：

* 全局变量和常量的定义
* 函数的定义
* 操作符和赋值
* 行内对齐
* 注释和文档描述块
* 类名的前缀及后缀
* 最佳实践

本规范之后的修订与扩展将弥补以上不足。

## 附录 A. 问卷调查

为了编写本规范，小组制定了调查问卷，用来统计各成员项目的共同规范。 以下是此问卷调查的数据，在此供查阅。

### A.1. 问卷数据

url,http://www.horde.org/apps/horde/docs/CODING\_STANDARDS,http://pear.php.net/manual/en/standards.php,http://solarphp.com/manual/appendix-standards.style,http://framework.zend.com/manual/en/coding-standard.html,http://symfony.com/doc/2.0/contributing/code/standards.html,http://www.ppi.io/docs/coding-standards.html,https://github.com/ezsystems/ezp-next/wiki/codingstandards,http://book.cakephp.org/2.0/en/contributing/cakephp-coding-conventions.html,https://github.com/UnionOfRAD/lithium/wiki/Spec%3A-Coding,http://drupal.org/coding-standards,http://code.google.com/p/sabredav/,http://area51.phpbb.com/docs/31x/coding-guidelines.html,https://docs.google.com/a/zikula.org/document/edit?authkey=CPCU0Us&hgd=1&id=1fcqb93Sn-hR9c0mkN6m\_tyWnmEvoswKBtSc0tKkZmJA,http://www.chisimba.com,n/a,https://github.com/Respect/project-info/blob/master/coding-standards-sample.php,n/a,Object Calisthenics for PHP,http://doc.nette.org/en/coding-standard,http://flow3.typo3.org,https://github.com/propelorm/Propel2/wiki/Coding-Standards,http://developer.joomla.org/coding-standards.html  
voting,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,no,no,no,?,yes,no,yes  
indent\_type,4,4,4,4,4,tab,4,tab,tab,2,4,tab,4,4,4,4,4,4,tab,tab,4,tab  
line\_length\_limit\_soft,75,75,75,75,no,85,120,120,80,80,80,no,100,80,80,?,?,120,80,120,no,150  
line\_length\_limit\_hard,85,85,85,85,no,no,no,no,100,?,no,no,no,100,100,?,120,120,no,no,no,no  
class\_names,studly,studly,studly,studly,studly,studly,studly,studly,studly,studly,studly,lower\_under,studly,lower,studly,studly,studly,studly,?,studly,studly,studly  
class\_brace\_line,next,next,next,next,next,same,next,same,same,same,same,next,next,next,next,next,next,next,next,same,next,next  
constant\_names,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper,upper  
true\_false\_null,lower,lower,lower,lower,lower,lower,lower,lower,lower,upper,lower,lower,lower,upper,lower,lower,lower,lower,lower,upper,lower,lower  
method\_names,camel,camel,camel,camel,camel,camel,camel,camel,camel,camel,camel,lower\_under,camel,camel,camel,camel,camel,camel,camel,camel,camel,camel  
method\_brace\_line,next,next,next,next,next,same,next,same,same,same,same,next,next,same,next,next,next,next,next,same,next,next  
control\_brace\_line,same,same,same,same,same,same,next,same,same,same,same,next,same,same,next,same,same,same,same,same,same,next  
control\_space\_after,yes,yes,yes,yes,yes,no,yes,yes,yes,yes,no,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes,yes  
always\_use\_control\_braces,yes,yes,yes,yes,yes,yes,no,yes,yes,yes,no,yes,yes,yes,yes,no,yes,yes,yes,yes,yes,yes  
else\_elseif\_line,same,same,same,same,same,same,next,same,same,next,same,next,same,next,next,same,same,same,same,same,same,next  
case\_break\_indent\_from\_switch,0/1,0/1,0/1,1/2,1/2,1/2,1/2,1/1,1/1,1/2,1/2,1/1,1/2,1/2,1/2,1/2,1/2,1/2,0/1,1/1,1/2,1/2  
function\_space\_after,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no,no  
closing\_php\_tag\_required,no,no,no,no,no,no,no,no,yes,no,no,no,no,yes,no,no,no,no,no,yes,no,no  
line\_endings,LF,LF,LF,LF,LF,LF,LF,LF,?,LF,?,LF,LF,LF,LF,?,,LF,?,LF,LF,LF  
static\_or\_visibility\_first,static,?,static,either,either,either,visibility,visibility,visibility,either,static,either,?,visibility,?,?,either,either,visibility,visibility,static,?  
control\_space\_parens,no,no,no,no,no,no,yes,no,no,no,no,no,no,yes,?,no,no,no,no,no,no,no  
blank\_line\_after\_php,no,no,no,no,yes,no,no,no,no,yes,yes,no,no,yes,?,yes,yes,no,yes,no,yes,no  
class\_method\_control\_brace,next/next/same,next/next/same,next/next/same,next/next/same,next/next/same,same/same/same,next/next/next,same/same/same,same/same/same,same/same/same,same/same/same,next/next/next,next/next/same,next/same/same,next/next/next,next/next/same,next/next/same,next/next/same,next/next/same,same/same/same,next/next/same,next/next/next

### A.2. 问卷说明

indent\_type: 缩进类型. tab = "使用 tab 键一次", 2 or 4 = "空格的数量"

line\_length\_limit\_soft: 每行字符数量的“软”限制. ? = 不可辩或无作答, no 表示无限制.

line\_length\_limit\_hard: 每行字符数量的“硬”限制. ? = 不可辩或无作答, no 表示无限制.

class\_names: 类名称的命名. lower = 只允许小写字母, lower\_under = 下滑线分隔的小写字母, studly = StudlyCase 的驼峰风格.

class\_brace\_line: 类的开始花括号是与 class 关键字在同一行或是在其的下一行？

constant\_names: 类的常量如何命名? upper = 下划线分隔的大写字母.

true\_false\_null: 关键字 true、false 以及 null 是全部小写 lower 还是全部大写 upper?

method\_names: 方法名称如何命名? camel = camelCase, lower\_under = 下划线分隔的小写字母.

method\_brace\_line: 方法的开始花括号是与方法名在同一行还是在其的下一行？

control\_brace\_line: 控制结构的开始花括号是与声明在同一行还是在其的下一行？

control\_space\_after: 控制结构关键词后是否有空格？

always\_use\_control\_braces: 控制结构体是否都要被包含在花括号内？

else\_elseif\_line: else 或 elseif 与前面的结束花括号在同一行还是在其的下一行？

case\_break\_indent\_from\_switch: switch 语句中的 case 和 break 需要相对 switch 缩进多少次？

function\_space\_after: 函数调用语句中，函数名称与变量列表的左括号间是否有空格？

closing\_php\_tag\_required: 纯 PHP 代码的文件，是否需要 ?> 结束标签？

line\_endings: 选择哪种类型的行结束符？

static\_or\_visibility\_first: 声明一个静态方法时，static 是写访问修饰符前还是后？

control\_space\_parens: 控制结构里，左括号后以及右括号前是否有空格？yes = if ( $expr ), no = if ($expr).

blank\_line\_after\_php: PHP 开始标签后，是否需要一个空行？

class\_method\_control\_brace: 开始花括号在类、方法和控制结构的位置统计。

### A.3. 问卷统计结果

indent\_type:  
 tab: 7  
 2: 1  
 4: 14  
line\_length\_limit\_soft:  
 ?: 2  
 no: 3  
 75: 4  
 80: 6  
 85: 1  
 100: 1  
 120: 4  
 150: 1  
line\_length\_limit\_hard:  
 ?: 2  
 no: 11  
 85: 4  
 100: 3  
 120: 2  
class\_names:  
 ?: 1  
 lower: 1  
 lower\_under: 1  
 studly: 19  
class\_brace\_line:  
 next: 16  
 same: 6  
constant\_names:  
 upper: 22  
true\_false\_null:  
 lower: 19  
 upper: 3  
method\_names:  
 camel: 21  
 lower\_under: 1  
method\_brace\_line:  
 next: 15  
 same: 7  
control\_brace\_line:  
 next: 4  
 same: 18  
control\_space\_after:  
 no: 2  
 yes: 20  
always\_use\_control\_braces:  
 no: 3  
 yes: 19  
else\_elseif\_line:  
 next: 6  
 same: 16  
case\_break\_indent\_from\_switch:  
 0/1: 4  
 1/1: 4  
 1/2: 14  
function\_space\_after:  
 no: 22  
closing\_php\_tag\_required:  
 no: 19  
 yes: 3  
line\_endings:  
 ?: 5  
 LF: 17  
static\_or\_visibility\_first:  
 ?: 5  
 either: 7  
 static: 4  
 visibility: 6  
control\_space\_parens:  
 ?: 1  
 no: 19  
 yes: 2  
blank\_line\_after\_php:  
 ?: 1  
 no: 13  
 yes: 8  
class\_method\_control\_brace:  
 next/next/next: 4  
 next/next/same: 11  
 next/same/same: 1  
 same/same/same: 6