编程规范

Version 2.8

1 为什么需要规范？

规范最大的好处是为团队协作带来一个统一的标准，有利于提高阅读代码的效率（可读性），提高代码质量（效率、稳定性），以及较少常见错误。

1.1 规范必须誓死贯彻吗？

并非如此，编程开发是很具创造性的活儿，如果您觉得哪一条规范扼杀了您的创造性或严重影响了您的编码效率，就应该提出来，我们再好好磋商。
但是值得注意的是在团队还没有达成新的共识之前，请务必尽量符合规范，为己为人书写符合标准之代码。

2 基本规范

基本规范适用于各种编程语言，如无特别难以克服的难题，所有代码必须遵守本“基本规范”。

2.1 文件编码

UTF-8 withOUT BOM

2.2 换行格式

UNIX 格式，LF only withOUT CRLF

2.3 最大行长度

最大行长度应尽量符合80字传统，各IDE均有辅助线提示。

【注意！】

- 最大行长度仅是建议性的，实在有必要，可不必拘泥；
- HTML 代码可无视此规定。

2.4 缩进规范

如无特别说明，所有下级嵌套代码需要往右缩进4个空格，不允许使用制表符（Tab）缩进。

【TIPS】

- 可在NetBeans（或其他 IDE ）中设置“按一下 Tab 按钮”为输出“4个空格缩进”（反之：“Shift + Tab”为撤销一级缩进）以提高书写代码的效率。
- HTML代码可自行处理代码缩进。

2.5 超链接及 **URL** 规范

1. 站内链接须以 / 开头，即指定由网站根目录到相应“控制器/动作/参数”的详细路径，不允许简写；
2. 控制器名后 须 有 /；
3. 如果动作名后没有进一步的参数， 勿 在末尾加 /；
4. 如果访问某个控制器的动作，并且动作后面还包含更深的路径参数，那么 URL 结尾的 / 可根据语义或功能逻辑自行判断。

示例：

```
href="/控制器/"
href="/控制器/动作"
href="/控制器/动作/参数1/参数2"
href="/控制器/动作/参数a/参数b/"
```

2.6 代码文件必须以新空白行结尾

也就是在所有代码结束后都必须加上一个额外的换行符结尾。

3 PHP

3.1 代码标识规范

- 嵌入PHP代码必须统一使用"<?php ... ?>"作为开始和结束标识，禁止使用如"<? ... ?>","<?=$var?>"或"<% ... %>"这样的短标识风格；
- 如果一个纯PHP代码文件（如类文件），而不是在HTML中嵌入PHP的文件，只需要在代码开头使用开始标识"<?php"，禁止在代码末尾添加结束标志"?>"。

3.2 代码文件规范

1. 所有包含PHP代码的文件统一使用".php"（小写）后缀，不允许使用其他兼容后缀；
2. 如非必要，代码文件不允许直接存放在"wwwroot"文件夹内（入口脚本或其他必要的情况除外）；
3. 文件名仅可以包含大写字母、小写字母、数字、下划线、中横线，其他符号严禁使用；
4. 一个类文件必须包含且仅包含一个类（或一个抽象类）；
5. 类文件以其所包含的类命名（文件名与类名在字母上必须完全一致），但类文件名必须全是小写字母（类名大小写按类命名规范为准）；
6. 类文件的名称空间和目录文件结构必须完全一致；
7. PHP代码文件必须包含文件级注释，该注释必须符合PHPDocumentor标准；
8. 类文件必须包含文件级注释和类注释，该注释必须符合PHPDocumentor标准；
9. 如非必要，不应该在类文件中包含其他额外的PHP代码，如果实在需要这样做，需要保证类代码和额外的PHP代码之间有两行或以上的分隔。

3.3 运算规范

3.3.1 数学符号、赋值符号、比较符号、连接符号等前后都需要加空格

```php
<?php
// 不推荐写法
$c=$a+$b;
if ($d===$e) {
    ...
}
$f='Leask'.'Huang';

// 推荐写法
$c = $a + $b;
if ($d === $e) {
    ...
}
$f = 'Leask' . 'Huang';
```

3.3.2 如无特殊情况，我们优先使用恒等符号"==="代替相等符号"=="做判断

```php
<?php
// 不推荐写法
if ('Apple' == 'Apple') {
    ...
}

// 推荐写法
if ('Apple' === 'Apple') {
    ...
}
```

【TIPS】相应地，我们优先使用不恒等符号"!=="代替不相等符号"!="做判断

3.3.3 禁止“内嵌赋值运算 试图提高运行效率”

```php
<?php
// 不推荐写法:
$d = ($a = $b + $c) + 7;

// 推荐写法:
$a = $b + $c;
$d = $a + 7;
```

3.3.4 不推荐一个语句中给多个变量赋相同的值

```php
<?php
// 不推荐写法:
$a = $b = 7;
```

3.3.5 不要将赋值运算符用在容易与相等关系运算符混淆的地方

```php
<?php
// 不推荐写法
if ($c++ = $d++) {
    ...
}

// 推荐写法
if (($c++ = $d++)) {
    ...
}
// 或
if (($c++ = $d++) !== 0) {
    ...
}
```

3.3.6 适当增加括号优化可读性

```php
<?php
$a = 1;
$b = 1;
$c = 2;
$d = 2;

if ($a === $b && $c === $d) $e = '不推荐写法';

if (($a === $b) && ($c === $d)) $e = '推荐写法';
```

3.3.7 三元运算符的断行

```php
<?php
$a = (aLongBooleanExpression) ? 'Yes' : 'No';

$b = (aLongBooleanExpression) ? 'Yes'
                              : 'No';

$c = (aLongBooleanExpression)
   ? 'Yes'
   : 'No';
```

3.4 字符串规范

3.4.1 尽量用"echo"代替"print"

```php
<?php
// 不推荐写法
print('Hello' . ' ' . 'World!');

// 推荐写法
echo 'Hello' , ' ' , 'World!';
```

3.4.2 尽量用","链接需要echo的字符串

```php
<?php
// 可接受的写法
echo 'Hello' . ' ' . 'World!';

// 推荐写法
echo 'Hello' , ' ' , 'World!';
```

3.4.3 如果一个字符串问题可以使用单引号解决，那么尽量不要使用双引号

```php
<?php
// 不推荐写法
$str = "Hello $name, welcome to the real world.";

// 推荐写法
$str = 'Hello ' . $name . ', welcome to the real world.';

// 确实需要使用双引号的情况下，应注意内嵌变量的表示方法

// 不推荐的写法
$str = "Hello ${name}, welcome to the real world.";

// 可接受的写法
$str = "Hello $name, welcome to the real world.";
$str = "Hello {$name}, welcome to the real world.";
```

[补丁] 在代码书写中理应尽量使用单引号替代双引号，但是结合到实际情况，若我们需要拼装比较复杂的字符串（如SQL语句）时。如果使用双引号能带来很大的便利，那么可以酌情使用。

3.4.4 如因赋值字符串较长，需要分行连接时，分行后的连接符号"."需要对齐该赋值语句的赋值符号"="

```
1 <?php
2 // 推荐写法
3 $str = 'Leask Huang loves '
4      . 'PHP, Javascript '
5      . 'and so on.';
```

3.5 数组规范

3.5.1 禁止使用负数作为数组索引

```
1 <?php
2 $arrayA[-7] = '不推荐写法';
3
4 $arrayB[7] = '推荐写法';
```

3.5.2 直接创建数组的时候，每个元素之间（逗号","之后），应该用空格分隔开

```
1 <?php
2 $arrayA = array(1,2,3,'不推荐','写法');
3
4 $arrayB = array(1, 2, 3, '推荐', '写法');
```

3.5.3 当数组的元素较多，需要断行时，可采用以下两种方式

```
 1 <?php
 2 $arrayA = array(1, 2, 3, '推荐', '写法一',
 3                 4, 5, 6,
 4                 7, 8, 9);
 5
 6 $arrayB = array(
 7     1, 2, 3, '推荐', '写法二',
 8     4, 5, 6,
 9     7, 8, 9,
10 );
```

【TIPS】使用第二种方式时，推荐在最后一个元素后加上一个逗号，这样能避免添加一行新元素后漏加逗号的常见错误。

3.5.4 优雅地创建关联数组

关联数组的大致规范和数字索引数组一致（如上例），与之有别的是推荐每行只放一个元素('键'=>'值')。
还有一点需要注意的是，每行的"=>"符号应该是相互对齐的，对齐规则是先处理“最长键名”的行，该行"=>"符号左右各留一个空格，其他行以该行为标准对齐。

```
1 <?php
2 $arrayA = array('firstKey'  => 'Leask',
3                 'secondKey' => 'Huang');
4
5 $arrayA = array(
6     'firstKey'  => 'Leask',
7     'secondKey' => 'Huang',
8 );
```

3.5.5 控制结构规范

```
 1 <?php
 2
 3 // if 语句基本结构规范
 4 if空($varA空===空'Apple')空{
 5     ... // 程序体必须在代码块中
 6 }空else空{
 7     ... // 程序体必须在代码块中
 8 }
 9
10 // if 语句当条件需要断行时的结构规范
11 if (($a === $b)
12     && ($b === $c) // 对齐if的第一个条件
13     || ($e === $f)
14 ) {
15     echo 'A';
16 } elseif (($a !== $b)
17           || ($b !== $c) // 对齐elseif的第一个条件
18 ) {
19     echo 'B';
20 } else {
```

```
    echo 'C';
}

// switch结构规范
switch空($varA)空{
    case空1:
        ...
        break;
    case空2:
        ...
        break;
    default:
        ...
        break;
}
```

【注意！】

1. 一般情况下应该尽量避免使用臭名昭著的goto语法，实在能提供巨大便利而又不影响程序结构的时候才允许使用；
2. 使用switch结构时不允许省略default一节；
3. 控制结构的程序体必须在语句块中，即使代码只有一句，也不允许单独存在，如下例。

```
<?php

// 推荐写法
if ($a === $b) {
    ...;
}

// 不推荐写法
if ($a === $b) { ...; }

// 不推荐写法
if ($a === $b) ...;
```

3.6 命名规范

3.6.1 变量命名规范

贯彻驼峰式变量命名风格，同时有以下几点补充：

1. 变量名可以是大写字母或小写字母，必要时也可以插入数字（仅存在于名称中间或结尾），但是不鼓励这样做；
2. 不以下划线或中横线链接单词；
3. 变量名以3个小写字母的类型简写作为固定前缀；
4. 前缀后每个单词的首字母大写，除首字母外其他字母均小写（即使是缩写词也应该如此，如HTML这个缩写词在以下变量名中：$strReturnHtml）；
5. 类中的私有变量（以"private"或"protected"定义）名应该以下划线"_"开头；
6. 尽量避免使用单字母或毫无意义的字母组合作为变量名，除非是作为短循环的循环变量（如$i, $j, $k...）；
7. 当一个循环大约超过20行代码的时候，就应该考虑为循环变量取更有意义的变量名。

```
<?php
// 整型数
$intDaysBeforeTwentyTwelve = 932;

// 浮点型
$flDaysBeforeTwentyTwelve = 932.777;

// 布尔型
$blnIsTwentyTwelveTrue = true;

// 字符串
$strNameOfTwentyTwelve = 'The End of the World!';

// 数组
$arrWhoDieInTwentyTwelve = array('Leask Huang', 'Bill Gates', 'Steve Jobs');

// 对象（类的实例）
$objNoahsArk = new SpaceShip;

// 资源（句柄）
$resDBConnect = database_connect(); // 假定"database_connect()"为数据库连接函数，返回一个数据库连接资源

// NULL型
$nllAfterTwentyTwelve = null;
```

3.6.2 常量命名规范

1. 常量名必须统一使用大写字母或数字；
2. 常量名各个单词之间需要通过下划线"_"连接；
3. 可以直接在PHP代码中定义常量，但是并不推荐这样做，比较好的做法是将其封装到类中。

3.6.3 函数（方法）命名规范

函数名遵守驼峰式命名风格，同时有以下几点需要补充：

1. 函数名只能包含大写字母、小写字母、必要时也可以使用数字（仅存在于名称中间或结尾），但是不鼓励这样做；
2. 如果某方法以"private"或"protected"定义，那么该方法名应以下划线"_"开头，其他情况禁止在函数（方法）名称中使用下划线；
3. 函数名的第一个单词首字母小写，其余单词首字母大写，除此外其他字母均小写（即使是缩写词也应该如此，如HTML这个缩写词在以下函数名中："outputHtml()"）；
4. 在面向对象编程时，设置或读取属性值的方法应该始终以"set"或"get"开头；
5. 可以直接在PHP代码中创建一个函数，但是并不推荐这样做，比较好的做法是将其封装为一个静态方法。

3.6.4 类命名规范

1. 类名应以大写字母开头；
2. 类名可以出现大写字母，小写字母和数字，之间可以用下划线相连，也可以直接相连；
3. 类名中的各个单词首字母应该大写，除首字母外其他字母均小写（即使是缩写词也应该如此，如HTML这个缩写词在以下类名中：WriteHtml）；
4. 抽象类类名需要添加"_Abstract"结尾。

```php
<?php
class SpaceShip
{
    ...
}

abstract class SpaceShip_Base_Abstract
{
    ....
}
```

3.7 函数规范

3.7.1 函数定义规范

```php
<?php
// 常规函数定义
function eat($argA, $argB, $argC)
{
    ...
}

// 参数断行处理
function drink($argD, $argE, $argF,
    $argG, $argH, $argI
) {
    ...
}

// 返回值例子
function returnExampleA()
{
    return('不推荐' . '写法');
}
function returnExampleB()
{
    return '推荐' . '写法';
}
```

3.7.2 函数（方法）调用规范

```php
<?php
// 调用一个函数的时候，各个参数之间应该用空格分隔
callFunctionExampleA(1, 2, 3);

// 当包含数组作为参数时
callFunctionExampleB(array(1, 2, 3), 4, 5);

// 当参数数组需要分行时,风格一
callFunctionExampleC(array(1, 2, 3, 4, 5,
                     6, 7, 8,
                     9, 'A', 'B'), 'C', 'D');

// 当参数数组需要分行时,风格二
callFunctionExampleD(array(
    1, 2, 3, 4, 5,
    6, 7, 8,
    9. 'A', 'B', 'C'
), 'D', 'E');
```

【注意！】类方法应始终以**"private"**、**"protected"**或**"public"**关键字定义；

3.8 类规范

```php
<?php
// 常规类定义
class SpaceShip
{
    ...
}

// 类定义换行规范，风格一
class NoahsArkA extends SpaceShip implements InterfaceA, InterfaceB
{
    ...
}

// 类定义换行规范，风格二
class NoahsArkB
    extends SpaceShip
    implements InterfaceC,
               InterfaceD
{
    ...
}
```

【注意！】

1. 类中任何变量声明都应放在类的顶部，在所有方法之前；
2. 如非必要不要使用var语法声明变量，变量应尽可能以"private"、"protected"或"public"形式声明。

3.9 注释规范

所有注释都需要符合phpDocumentor规范，详情可参见《PhpDocumentor 快速入门》。

以下是 南京安元科技有限公司 联合研发中心 SaaS组 标准文件注释块：

```php
<?php
/**
 * 此文件的标题或简介
 *
 * 此文件的的详细简介，如无详细简介可不写。
 *
 * @category   AYSaaS
 * @package    AYSaaS_Example_Pack
 * @copyright  Copyright (C) 2010 AnYuan-SaaS (http://www.aysaas.com)
 * @author     Your Name <*******@gmail.com>
 * @version    0.7 beta
 * @link       http://www.aysaas.com
 * @since      0.3 Alpha
 */
```

【TIPS】仅供内部查阅的单行注释（直接在双斜线"//"后添加的注释）是允许的，我们推荐在必要的地方添加友好的小注释，这将大大改善程序的可读性，而且有做笔记、写小TIPS的功能，使代码对团队成员更加友好，也防止自己会遗忘程序的关键信息。

4 SQL

本章主要探讨PHP中使用MySQL时应遵守的规范。

4.1 编码规范

所有数据库和表，统一使用UTF-8编码。
在程序执行和传递数据的过程中，始终贯彻UTF-8编码。

```sql
CREATE DATABASE db_test CHARACTER SET UTF8;
```

4.2 命名规范

1. 所有的数据库名、表名和列名都应该尽量使用小写字母命名；

2. 尽量取有意义，便于理解和记忆的数据库名、表名和列名；
3. 如果需要使用有多个单词，各个单词之间应该用下划线链接，如"tbl_user_rights"；
4. 表名可以用单数或复数形式，但是同一层次的单复数取舍应该是一致的，不能一会是"table_users"（复数），一会是"table_company"（单数），这样很容易让人误解，影响可读性；

4.3 建表规范

1. 所有开发阶段需要建立的数据库和表都需要保留原SQL语句文件，如"tbl_var.sql"，便于项目部署；
2. 如无特殊说明，建表时使用"InnoDB"作为默认的存储引擎。

```
CREATE TABLE tbl_var (
    id INTEGER NOT NULL PRIMARY KEY AUTO_INCREMENT,
    logid INTEGER NOT NULL,
    logname VARCHAR(256) NOT NULL,
    logvalue VARCHAR(256) NOT NULL
) ENGINE = INNODB;
```

4.4 SQL语句规范

1. SQL关键字必须大写；
2. 表名、列名必须用"`"括起来；
3. 语句断行后连接符号"."需要和该语句的赋值符号"="对齐。

```
<?php
// 不推荐写法
$strSqlA = "select id, name from tbl_people "
    . "where name='Bill' or name='Steve'";

// 推荐写法
$strSqlB = "SELECT `id`, `name` from `people` "
          . "WHERE `name`='Bill' OR `name`='Steve'";
```

5 HTML

在保证IE8兼容的前提下遵守HTML5规范

5.1 基本文档模型

DOCTYPE和META应如下声明:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Example document</title>
  </head>
  <body>
    <p>Example paragraph</p>
  </body>
</html>
```

【TIPS】HTML5规范的具体细节可参考： http://www.w3.org/TR/html5-diff/

5.2 单独标签无须关闭符号

单个出现（而非成对出现）的标签，无须添加关闭符号（"/"）。
留意下例中的"<meta...>"与"
"标签，无须写成"<meta.../>"与"
"的形式：

```
...
    <meta charset="UTF-8">
...
    <p>Example paragraph<br>Example paragraph</p>
...
```

6 Javascript

Javascript 语法过于灵活，不作规范难以保证编码质量，甚至影响执行效率。
我们主要参考比较成熟的《Google JavaScript Style Guide》作为规范，请大家自行学习：
http://google-styleguide.googlecode.com/svn/trunk/javascriptguide.xml

7 CSS

7.1 基本原则

1. 所有对页面样式、对齐方式等外观层面上的控制都应通过CSS方式控制，禁止使用HTML样式（这种方式已渐被淘汰）；
2. 所有CSS代码尽可能存放在独立的"*******.css"文件中，尽可能不要在HTML中穿插CSS代码块，更不要使用行内样式（“ ... ”）；
3. 大体上执行CSS 2.1标准，但若某CSS 3新功能在 Internet Explorer 8+ 和 Mozilla Firefox 3.6+ 都支持良好，则可考虑引入；
4. 尽可能把相关的CSS样式分为一组，并简要注释，见下文 第七章——第四节（7.5） 中的例子。

7.2 载入规范

```
1 <link rel="stylesheet" href="/global/styles/base.css" type="text/css" />
```

7.3 标准格式

```
1 /* 选择器与样式之间，样式与花括号"{","}"之间，以及样式与样式之间都需要用空格分开：  */
2 html,body,div空{空margin:空0;空padding:空0;空}
3
4 /* 即：  */
5 html,body,div { margin: 0; padding: 0; }
```

7.4 超链接样式顺序

超链接样式必须按照以下顺序定义

```
1 a:link { ... }
2 a:visited { ... }
3 a:hover { ... }
4 a:actived { ... }
```

7.5 换行规范

1. 如果样式比较短，可以一行定义样式，如下例中的"/* REST */"一节；
2. 如果样式比较长，推荐每行一个属性，并缩进4个空格，如下例中的"/* LAYOUT */"一节。

```
 1 /* RESET */
 2 html,body,div,ul,ol,li,dl,dt,dd,h1,h2,h3,h4,h5,h6,pre,form,p,blockquote,fieldset,input { margin: 0; padding: 0; }
 3 h1,h2,h3,h4,h5,h6,pre,code,address,caption,cite,code,em,strong,th { font-size: 1em; font-weight: normal; }
 4 ul,ol { list-style: none; }
 5 fieldset,img { border: none; }
 6 caption,th { text-align: left; }
 7 table { border-collapse: collapse; border-spacing: 0; }
 8
 9 /* LAYOUT */
10 .clearer {
11     clear: both;
12     display: block;
13     margin: 0;
14     padding: 0;
15     height: 0;
16     line-height: 1px;
17     font-size: 1px;
18 }
```

8 其他参考资料

8.1 常用简写

编程中需要命名变量、常量、函数和类时可以根据需要对某些单词进行缩写，常用的规范缩写有：

```
initialization > init
temp           > tmp
flag           > flg
statistic      > stat
increment      > inc
message        > msg
column         > col
```

如果您需要使用一个只有自己才能看懂的缩写词，那么请在注释处加以说明，如：

```
leask > lsk
```

8.2 常用的反义（相对）词对

编程中需要命名一些相反或相对的变量、常量、函数和类时可以根据需要参考以下常用词对：

```
add       / remove
begin     / end
create    / destroy
insert    / delete
first     / last
get       / release
increment / decrement
put       / get
add       / delete
lock      / unlock
open      / close
min       / max
old       / new
start     / stop
next      / previous
source    / target
show      / hide
send      / receive
source    / destination
cut       / paste
up        / down
```

8.3 常用的页面布局元素

做前端HTML+CSS页面布局的时候经常需要为页面的某布局元素命名，这里提供一些常用的元素名，供大家参考：

```
header
content
container
footer
nav
sidebar
column
top-panel
horizontal-nav
left-side / right-side
left-col / right-col
center-col
branding
main-nav
main-content
```