****

**HƯỚNG DẪN PHONG CÁCH CODING**

**Thực hiện : Nguyễn Đăng Khoa & Trịnh Đình Phúc**

**MỤC LỤC**

[**NỘI DUNG 1: TỔNG QUAN** 3](#_Toc512507120)

[**NỘI DUNG 2: KHÁI QUÁT CHUNG** 5](#_Toc512507121)

[**2.1 Tiêu chuẩn mã hóa cơ bản** 5](#_Toc512507122)

[**2.2 Các tập tin** 5](#_Toc512507123)

[**2.3 Dòng** 5](#_Toc512507124)

[**2.4 Thụt lề** 5](#_Toc512507125)

[**2.5 Các từ khóa True, False và Null** 5](#_Toc512507126)

[**NỘI DUNG 3: LỚP, THUỘC TÍNH, PHƯƠNG THỨC** 6](#_Toc512507127)

[**3.1 Implements và Exstends** 6](#_Toc512507128)

[**3.2 Thuộc tính** 7](#_Toc512507129)

[**3.3 Phương thức** 7](#_Toc512507130)

[**3.4 Đối số của phương thức** 7](#_Toc512507131)

[**3.5 Abstract, final, static** 9](#_Toc512507132)

[**3.6 Phương thức và hàm** 9](#_Toc512507133)

[**NỘI DUNG 4: CẤU TRÚC ĐIỀU KHIỂN** 10](#_Toc512507134)

[**4.1 if, else if , else** 10](#_Toc512507135)

[**4.2 Switch case** 10](#_Toc512507136)

[**4.3 While, do while** 11](#_Toc512507137)

[**4.4 For** 11](#_Toc512507138)

[**4.5 Foreach** 11](#_Toc512507139)

[**4.6 Try , Catch** 12](#_Toc512507140)

[**NỘI DUNG 5: QUI ĐỊNH ĐÓNG MỞ NGOẶC** 13](#_Toc512507141)

[**NỘI DUNG 6 : KẾT LUẬN** 15](#_Toc512507142)

# **NỘI DUNG 1: TỔNG QUAN**

Mục đích của hướng dẫn này là giảm rủi ro khi xây dựng code từ các tác giả khác nhau. Nó làm như vậy bằng cách liệt kê một bộ chia sẻ các quy tắc và kỳ vọng về cách định dạng mã PHP.

Các quy tắc về phong cách ở đây có nguồn gốc từ sự tương đồng giữa các dự án thành viên khác nhau . Khi các tác giả khác nhau cộng tác trên nhiều dự án, nó giúp có một bộ nguyên tắc được sử dụng trong tất cả các dự án đó. Vì vậy, lợi ích của tài liệu hướng dẫn này không nằm trong các quy tắc, mà là trong việc chia sẻ các quy tắc đó.

PSR có nghĩa là PHP Standards Recommendations, nó là tiêu chuẩn được khuyến nghị áp dụng khi lập trình PHP và được các lập trình viên, tổ chức chấp nhận sử dụng. PSR được soạn thảo, đánh giá và khuyến khích sử dụng bởi một nhóm chuyên gia PHP những người phát triển cho các Framework và hệ thống PHP phổ biến. Bài viết này xin được trình bày theo chuẩn PSR-1.

Một số qui định trong chuẩn PSR-1:

* Code phải tuân theo tiêu chuẩn coding PSR-1.
* Code phải dùng 4 dấu cách để thụt lề, không phải Tabs.
* Độ dài tối đa của một dòng là 120 kí tự, nhưng nên để 80 kí tự hoặc thấp hơn.
* Sau Namespace nên là một dòng trắng, và bắt buộc để một dòng trắng sau lệnh use.
* Mở ngoặc của một class phải là ở dòng kế tiếp và đóng ngoặc của class phải là dòng tiếp theo ngay sau body.
* Mở ngoặc của một phương thức phải là dòng kế tiếp và đóng ngoặc của phương thức phải là dòng tiếp theo ngay sau body.
* Visibility phải được định nghĩa trên các thuộc tính và các phương thức. Abstract và final phải được định nghĩa trước Visibility. Static phải được định nghĩa sau Visibility.
* Cấu trúc điều khiển các keyword phải có một khoảng trắng ngay sau nó.
* Mở ngoặc của cấu trúc điều khiển cần nằm trên cùng một dòng, đóng ngoặc ở dòng kế tiếp sau body

**// Ví dụ tổng quan chuẩn PSR-1**

**<?php**

**namespace** Vendor\Package;

**use** FooInterface;

**use** BarClass **as** Bar;

**use** OtherVendor\OtherPackage\BazClass;

**class** **Foo** **extends** Bar **implements** FooInterface

{

**public** **function** **sampleMethod**($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*

}

}

# **NỘI DUNG 2: KHÁI QUÁT CHUNG**

## **2.1 Tiêu chuẩn mã hóa cơ bản**

* Mã PHẢI tuân thủ tất cả các quy tắc được nêu trong PSR-1 .

## **2.2 Các tập tin**

* Tất cả các tệp PHP PHẢI sử dụng dòng Unix LF (linefeed) kết thúc.
* Tất cả các tệp PHP PHẢI kết thúc bằng một dòng trống duy nhất.
* ?>Thẻ đóng phải được bỏ qua từ các tệp chỉ chứa PHP.

## **2.3 Dòng**

* KHÔNG được có một giới hạn cứng về độ dài của dòng.
* Giới hạn mềm trên chiều dài dòng PHẢI là 120 ký tự.
* Dòng KHÔNG NÊN dài hơn 80 ký tự; dòng dài hơn NÊN được chia thành nhiều dòng tiếp theo không quá 80 ký tự.
* KHÔNG được có khoảng trống cuối ở cuối dòng không trống.
* Các dòng trống CÓ THỂ được thêm vào để cải thiện khả năng đọc và chỉ ra các khối mã liên quan .
* KHÔNG được có nhiều hơn một câu trên mỗi dòng.

## **2.4 Thụt lề**

* Mã PHẢI sử dụng thụt lề 4 khoảng trắng và KHÔNG ĐƯỢC sử dụng các tab để thụt lề.
* Chỉ sử dụng dấu cách và không trộn các dấu cách với các tab, giúp tránh các sự cố với các khác biệt, bản vá, lịch sử và chú thích

## **2.5 Các từ khóa True, False và Null**

* Từ khóa PHP PHẢI nằm trong trường hợp thấp hơn.
* Các hằng PHP true, falsevà null PHẢI được trong trường hợp thấp hơn.

# **NỘI DUNG 3: LỚP, THUỘC TÍNH, PHƯƠNG THỨC**

## **3.1 Implements và Exstends**

- Các từ khóa extendsvà implementsphải được khai báo trên cùng một dòng với tên lớp.

- Cú pháp của lớp phải đặt trên 1 dòng riêng của nó

**<?php**

**namespace** Vendor\Package;

**use** FooClass;

**use** BarClass **as** Bar;

**use** OtherVendor\OtherPackage\BazClass;

**class** **ClassName** **extends** ParentClass **implements** \ArrayAccess, \Countable

{

*// constants, properties, methods*

}

* Danh sách implementsCÓ THỂ được chia thành nhiều dòng, trong đó mỗi dòng tiếp theo được thụt vào một lần. Khi làm như vậy, mục đầu tiên trong danh sách PHẢI nằm trên dòng tiếp theo, và PHẢI chỉ có một nội dung trên mỗi dòng.

**<?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*

}

## **3.2 Thuộc tính**

* Các từ khóa khai báo không phải để sử dụng khai báo một giá trị. Không phải có nhiều hơn một thuộc tính được khai báo trên mỗi câu lệnh. Tên thuộc tính KHÔNG được đặt trước bằng dấu gạch dưới để cho biết proteced hay private.

**<?php**

**namespace** Vendor\Package;

**class** **ClassName**

{

**public** $foo **=** **null**;

}

## **3.3 Phương thức**

* Tên phương thức KHÔNG NÊN được đặt trước bằng dấu gạch dưới đơn để cho biết khả năng hiển thị được proteced hay private.
* Tên phương thức KHÔNG được khai báo với dấu cách sau tên phương thức. Cú mở đầu PHẢI đi trên dòng riêng của nó, và cú đúp đóng phải đi trên dòng tiếp theo sau cơ thể. KHÔNG được có khoảng trống sau dấu ngoặc đơn mở , và KHÔNG có khoảng trống trước dấu ngoặc đơn đóng.

**<?php**

**namespace** Vendor\Package;

**class** **ClassName**

{

**public** **function** **fooBarBaz**($arg1, **&**$arg2, $arg3 **=** [])

{

*// method body*

}

}

## **3.4 Đối số của phương thức**

* Danh sách số không được có một khoảng cách trước dấu phẩy và phải có một khoảng cách sau dấu phẩy.
* Đối số có giá trị mặc định phải nằm cuối danh sách đối số

**<?php**

**namespace** Vendor\Package;

**class** **ClassName**

{

**public** **function** **foo**($arg1, **&**$arg2, $arg3 **=** [])

{

*// method body*

}

}

* Danh sách đối số CÓ THỂ được chia thành nhiều dòng, trong đó mỗi dòng tiếp theo được thụt vào một lần. Khi làm như vậy, mục đầu tiên trong danh sách PHẢI nằm trên dòng tiếp theo và chỉ PHẢI có một đối số trên mỗi dòng.
* Khi danh sách đối số được phân tách trên nhiều dòng, dấu ngoặc đơn đóng và dấu ngoặc mở phải được đặt cùng nhau trên một dòng riêng của chúng với một dấu cách giữa chúng.

**<?php**

**namespace** Vendor\Package;

**class** **ClassName**

{

**public** **function** **aVeryLongMethodName**(

ClassTypeHint $arg1,

**&**$arg2,

**array** $arg3 **=** []

) {

*// method body*

}

}

## **3.5 Abstract, final, static**

* Khi có sự xuất hiện của các cú pháp abstract và final PHẢI đứng trước khai báo khả năng hiển thị. Static phải nằm sau visibility declaration.

**<?php**

**namespace** Vendor\Package;

**abstract** **class** **ClassName**

{

**protected** **static** $foo;

**abstract** **protected** **function** **zim**();

**final** **public** **static** **function** **bar**()

{

*// method body*

}

}

## **3.6 Phương thức và hàm**

* Khi thực hiện một phương thức hay gọi hàm thì không có dấu cách giữa bên trong bên ngoài dấu ngoặc đơn. Khi phương thức và hàm cần thực hiện có đối số thì nó cần được truyền vào theo qui cách đã thực hiện ở phần đối số nêu trên, các đối số phải cần được truyền vào theo đúng thứ tự khai báo.

**<?php**

bar();

$foo**->**bar($arg1);

Foo**::**bar($arg2, $arg3);

* Danh sách đối số CÓ THỂ được chia thành nhiều dòng, trong đó mỗi dòng tiếp theo được thụt vào một lần. Khi làm như vậy, mục đầu tiên trong danh sách PHẢI nằm trên dòng tiếp theo và chỉ PHẢI có một đối số trên mỗi dòng.

**<?php**

$foo**->**bar(

$longArgument,

$longerArgument,

$muchLongerArgument

);

# **NỘI DUNG 4: CẤU TRÚC ĐIỀU KHIỂN**

- Các quy tắc chung cho các cấu trúc điều khiển như sau:

* Phải có một khoảng trống sau từ khóa cấu trúc điều khiển
* Không có một khoảng trống sau dấu ngoặc đơn mở
* Không có một khoảng trống trước dấu ngoặc đơn đóng
* Phải có một khoảng trống giữa dấu ngoặc đóng và mở ngoặc
* Cơ cấu cấu trúc phải được thụt vào một lần
* Đóng ngoặc phải PHẢI trên dòng tiếp theo sau khi nội dung
* Phần nội dung của mỗi cấu trúc PHẢI nằm trong dấu ngoặc nhọn. Điều này tiêu chuẩn hóa các cấu trúc và giảm khả năng lỗi khi các dòng mới được thêm vào nội dung.

## **4.1 if, else if , else**

* Một if cấu trúc được thể hiện như sau

**<?php**

**if** ($expr1) {

*// if body*

} **elseif** ($expr2) {

*// elseif body*

} **else** {

*// else body;*

}

* Từ khóa elseif NÊN được sử dụng thay vì else if để các từ khóa được kiểm soát một cách đơn lẻ rõ ràng câu lệnh.

## **4.2 Switch case**

* Một cấu trúc Switch được trình bày như ví dụ bên dưới. Các casetuyên bố PHẢI được thụt vào một lần từ switch, và breaktừ khóa (hoặc từ khóa chấm dứt khác) phải được thụt vào ở mức độ tương tự như case nội dung

**<?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**;

}

## **4.3 While, do while**

- Một cú pháp while và do while được thê hiện như sau

**<?php**

**while** ($expr) {

*// structure body*

}

?>

**<?php**

**do** {

*// structure body;*

} **while** ($expr);

## **4.4 For**

- Một cú pháp for được thể hiện như sau

**<?php**

**for** ($i **=** 0; $i **<** 10; $i**++**) {

*// for body*

}

## **4.5 Foreach**

- Cú pháp Foreach được thể hiện như sau

**<?php**

**foreach** ($iterable **as** $key **=>** $value) {

*// foreach body*

}

## **4.6 Try , Catch**

- Cú pháp Try Catch được thể hiện như sau

**<?php**

**try** {

*// try body*

} **catch** (FirstExceptionType $e) {

*// catch body*

} **catch** (OtherExceptionType $e) {

*// catch body*

}

# **NỘI DUNG 5: QUI ĐỊNH ĐÓNG MỞ NGOẶC**

* Đóng mở ngoặc phải được khai báo bằng dấu cách sau từ khóa function và khoảng trắng trước và sau từ khóa use.
* Nẹp mở phải đi trên cùng một dòng, và cú đúp đóng phải đi trên dòng tiếp theo sau nội dung.
* KHÔNG được có một khoảng trống sau dấu ngoặc đơn mở của danh sách đối số hoặc danh sách biến, và KHÔNG được có khoảng trống trước dấu ngoặc đơn đóng của danh sách đối số hoặc danh sách biến.
* Trong danh sách đối số và danh sách biến, KHÔNG được có một dấu cách trước mỗi dấu phẩy và phải có một dấu cách sau mỗi dấu phẩy.
* Đối số đóng với giá trị mặc định PHẢI đi vào cuối danh sách đối số .

**<?php**

$closureWithArgs **=** **function** ($arg1, $arg2) {

*// body*

};

$closureWithArgsAndVars **=** **function** ($arg1, $arg2) **use** ($var1, $var2) {

*// body*

};

* Danh sách đối số và danh sách biến CÓ THỂ được chia thành nhiều dòng, trong đó mỗi dòng tiếp theo được thụt vào một lần. Khi làm như vậy, mục đầu tiên trong danh sách PHẢI nằm trên dòng tiếp theo và chỉ PHẢI có một đối số hoặc biến trên mỗi dòng.
* Khi danh sách kết thúc (cho dù đối số hoặc biến) được chia trên nhiều dòng, dấu ngoặc đơn đóng và dấu ngoặc mở phải được đặt cùng nhau trên một dòng riêng của chúng với một dấu cách giữa chúng.
* Sau đây là ví dụ về các bao đóng có và không có danh sách đối số và danh sách biến được chia thành nhiều dòng.

**<?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*

};

* Các quy tắc định dạng cũng áp dụng khi đóng được sử dụng trực tiếp trong một hàm hoặc gọi phương thức làm đối số.

**<?php**

$foo**->**bar(

$arg1,

**function** ($arg2) **use** ($var1) {

*// body*

},

$arg3

);

# **NỘI DUNG 6 : KẾT LUẬN**

Có nhiều yếu tố về phong cách và thực hành cố tình bỏ qua bởi   
hướng dẫn này . Chúng bao gồm nhưng không giới hạn:

* Khai báo các biến toàn cầu và các hằng số toàn cầu
* Khai báo chức năng
* Toán tử và nhiệm vụ
* Căn chỉnh liên tuyến
* Nhận xét và khối tài liệu
* Tiền tố tên lớp và hậu tố
* Thực hành tốt nhất

Các đề xuất trong tương lai CÓ THỂ sửa đổi và mở rộng hướng dẫn này để giải quyết những yếu tố đó hoặc các yếu tố khác về phong cách và thực hành.